The University of Alberta
Printing Department
Edmonton, Alberta

THE UNIVERSITY OF ALBERTA

RESOURCE SHARING IN COMPUTER NETWORKS

BY

© GARRY HELANDER

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDY AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA

FALL, 1974

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled Resource Sharing in Computer Networks submitted by Garry Helander in partial fulfilment of the requirements for the degree of Master of Science.

Date June

ABSTRACT

To achieve a sophisticated level of interaction, resource sharing among members of a computer network is investigated. Several assertions which explain the resource-sharing process are made. An empirical investigation, based on a probabilistic model of the process, confirms these assertions. Furthermore, several new and interesting properties of the resource-sharing process are revealed. These results are related to an executive system for resource sharing in computer networks.

ACKNOWLEDGEMENT

I wish to express appreciation to my supervisor, John Tartar, for his valuable encouragement and criticism throughout this work.

The financial assistance of the Department of Computing Science and National Research Council is gratefully acknowledged.

Table of Contents

Page

Chapter 1 Introduction 1

Chapter 2 A Review of Computer Networks 4

2.1 Network Organization 4

2.1.1 Network Topology 4

2.1.2 Network Control 7

2.1.3 Communication 8

2.1.4 Network Nodes 9

2.1.5 Logical Structure 11

2.2 Computer Networks 13

2.2.1 Current Networks 13

2.2.2 Network Problems 24

Chapter 3 The Resource-Sharing Problem 29

3.1 Introduction 29

3.2 Current Resource-Sharing Networks 31

3.3 The Resource-Sharing Process 32

3.3.1 A Resource-Sharing Executive System 33

3.3.2 A Resource-Sharing Model 34

3.4 Summary 37

Table of Contents (cont'd)

Page

Chapter 4 Resource Sharing 38

4.1 Introduction 38

4.1.1 The Resource-Sharing Model 38

4.1.2 Load on the Network 40

4.2 The Resource-Sharing Process 41

4.2.1 Principle of Network Multiplicity 41

4.2.2 Fragmentation 46

4.2.3 Total Resource Utilization 50

4.2.4 Local Resource Utilization 52

4.3 Resource Allocation in Computer Networks 60

4.3.1 Three Resource-Allocation Strategies 61

4.3.2 Limited-Acceptance Allocation 66

4.3.3 Dynamic Allocation 68

Chapter 5 Conclusions 71

5.1 Summary of Results 71

5.2 Application of Results 72

5.3 Suggestions for Further Work 73

5.3 Conclusion 75

Bibliography 76

Appendix A 88

List of Tables

Table		Page
4.1	Parameters for three simulation experiments.	45
4.2	Parameters of a simulation to examine resource utilization under varying loads.	48
4.3	Wait queue size for a moderate load.	55
4.4	Queue size and local utilization for a heavy load.	57
4.5	Parameters for a comparison of three resource control algorithms.	64
4.6	Parameters of a simulation for dynamic resource control.	69

List of Figures

Figure		Page
2.1	Network topologies.	6
4.1	Total resource utilization in a five node network with varying load.	47
4.2	Resource utilization in a five node network with varying load.	47
4.3	Percentage resource utilization in a five node network with varying load.	48
4.4	Total resource utilization in a multi-node network with varying load.	51
4.5	Total resource utilization versus load for a multi-node network.	51
4.6	Local resource utilization in a multi-node network with a light load.	54
4.7	Local resource utilization in a multi-node network with a moderate load.	54
4.8	Local resource utilization in a multi-node network with a heavy load.	59
4.9	Local resource utilization in a multi-node network with an over-load.	59
4.10	Total resource utilization for three resource control algorithms.	63
4.11	Local resource utilization (normalized) for three resource control algorithms.	63

List of Figures (cont'd)

Figure Page

4.12 Total resource utilization for the limited acceptance resource control algorithm. 67

4.13 Remote resource utilization for the limited acceptance resource control algorithm. 67

4.14 Local resource utilization for the dynamic resource control algorithm. 70

CHAPTER 1

INTRODUCTION

A computer network is a set of autonomous computer systems which communicate with each other. The purpose of their interconnection is to permit interactive resource utilization, to share programs and data bases, or to achieve reliability and load leveling.

A computer network may encompass a wide variety of users and serve many diverse purposes. It makes available to a large population a wide selection of hardware and software capabilities which, individually, they could not afford. From the user's viewpoint, a computer network can provide access to a reliable computer facility which has many specialized applications. To the computing-center management, the network offers attractive economic considerations as well as as expanding customer services. The systems programmer sees the network as providing specialized applications programs and hardware capabilities. Thus for a variety of reasons, unique to each user's requirements, a computer network presents attractive alternatives and opportunities over a conventional system.

A computer network encounters many problems if it is to function as a viable entity. Perhaps the most fundamental and obvious of these is communication. This involves not

only the physical means of communication but also a protocol which allows machines to interpret and acknowledge messages correctly. Coupled with communication is the question of network architecture. It is the design of the network which meets some required performance specifications.

In contrast to the former points, which consider the physical make-up of the network, the third problem area deals with the logical structure of a computer network. It is at this level that the utility of the network is defined. The network's organization and structure makes available to the user the resources, the data bases, and the software. However, a logical structure which supplies these facilities and yet remains practical and feasible is difficult to achieve.

In terms of the previous paragraph this thesis considers the logical structure and control of a computer network. The particular consideration is a study of resource sharing among members of a computer network. The objective is to attain resource sharing at a more sophisticated level than is currently available. To accomplish this a fundamental understanding of resource sharing is essential. This work is an examination of the underlying process involved in resource sharing. Assertions which explain the process are made and empirical results substantiating these

assertions are given. Several important conclusions which will have a great influence on both current and future resource sharing networks are drawn.

Subsequent chapters will discuss these points in greater detail. Chapter two presents a brief outline and review of all aspects of computer networks. There will be a discussion of current networks and some consideration given to areas of current research. Chapter three presents in greater detail the problem which this thesis considers. It outlines the restrictions and limitations under which this study takes place. Chapter four investigates the fundamental process of resource sharing in relation to a computer network. The final chapter summarizes the conclusions of the work and discusses further extensions.

In summary, computer networks represent a major part of computer services both currently and in the future. Their potential utility and serviceability comes from resource sharing. This study considers means of increasing the level of resource sharing beyond that which is currently practised.

CHAPTER 2

A REVIEW OF COMPUTER NETWORKS

A computer network is a set of autonomous computer systems which communicate with each other. The purpose of their interconnection is to permit interactive resource utilization, to share programs and data bases, or to achieve reliability and load sharing [1,2,3,5].

Conceptually a computer network may be visualized as a graph in which edges are communication links and nodes are computers, terminals, and other networks. In subsequent sections this primitive notion will be expanded to a more formal description. Following this, several current networks will be reviewed and relevant problems encountered in computer networks discussed.

2.1 NETWORK ORGANIZATION

In general there are five elements which define a network; topology, control, communication, nodes, and logical structure. The first four are the building blocks of any network while the last defines the functional capability of the network.

2.1.1 NETWORK TOPOLOGY

The topology of a network is the configuration of nodes

and links which characterize the network. There are essentially three types: (1) star, (2) distributed and (3) ring (see Figure 2.1). The star configuration is represented by a communication structure in which all inter-node traffic passes through a central switching node. This of course gives high utilization of the data links but is susceptible to saturation and sacrifices reliability [6]. In a distributed topology the inter-node links are more general. There is less sharing of circuits, with increased reliability and flexibility. This is a natural consequence of the greater connectivity of the network as a whole. If each node is directly linked to each other node, the network is said to be fully connected. However it is more common to provide at least two but not all possible paths between nodes. A ring network may be considered a subcase of a distributed network. However its unique communication structure serves to delineate the two classes. In a ring topology, communication is only in one direction and is usually multiplexed into slots which a node may "claim" and use to transmit messages.

In summary, the topology of a network deals only with the relative position of nodes in the network. It is of great importance in determining the reliability and performance of the network.

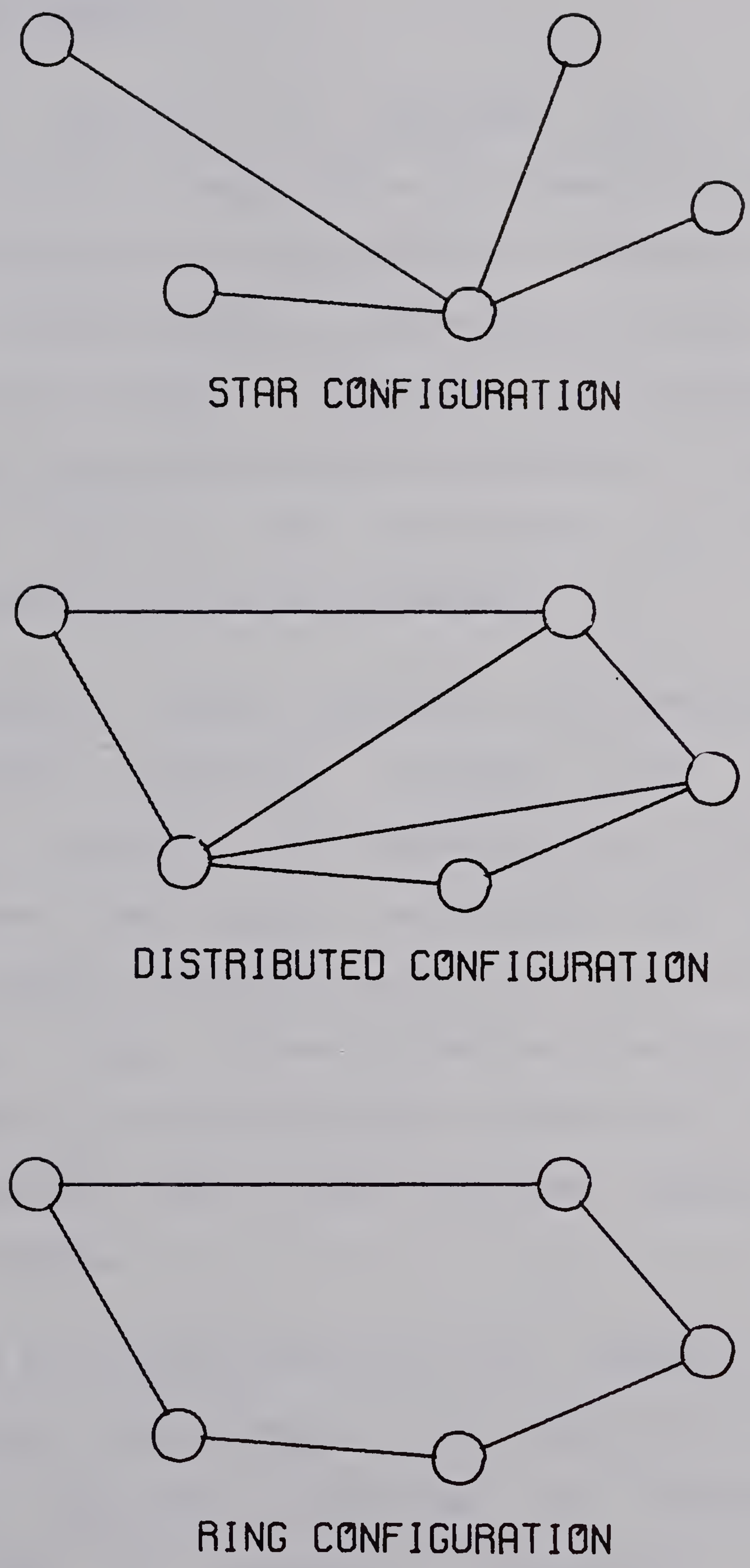

Figure 2.1 Network Topologies

2.1.2 NETWORK CONTROL

Network control is concerned with the overall performance of the network. There are essentially two types: (1) centralized and (2) distributed. A centralized control structure is one in which a single node exercises authority over the entire network. In contrast, a distributed control function has the control algorithm embedded in each node of the network. There is not necessarily any relationship between control structure and topology.

The primary control functions are: establishing node-to-node links; creating, breaking and maintaining connections; ensuring that messages are correct and in sequence when they arrive; monitoring and adapting to changes in network configurations due to failure of hosts or communication lines. Another important function is in flow control, that is, routing messages through the network to avoid congestion and provide graceful degradation of an overloaded system.

Part of the control function also includes monitoring and maintenance. Communication lines and node interfaces are tested at periodic intervals to determine their serviceability. In addition, statistics on network utilization and performance are taken for future analysis and improvements. A more static aspect of network control is

the designation of protocol -- that is, establishing a procedure by which hosts may communicate. This would include message addressing, format, and acknowledgements.

In summary, the jurisdiction of network control includes those functions which affect the smooth operation of the network entity.

2.1.3 COMMUNICATION

The communication element of a computer network is the physical means of information interchange. This may include digital or analog transmission on a wide variety of devices; twisted pair, coaxial cable, microwave, radio links [65], etc. Although a network may use many communication media, it is usually homogeneous in its use of channels, either message-switched or circuit-switched.

A circuit-switched channel is a direct link between source and destination for the duration of the message. Since the overhead can be large, this organization is useful only for long messages. In contrast, the more flexible message-switched channel is usually a digital link between nodes. A message is placed on the communication circuit and it is shunted from node to node until it arrives at the destination site. The intermediary nodes temporarily store the messages and then forward them to the next node when the

communication channel is free -- hence the term "store-and-forward". This format is efficient for short interactive messages but can be cumbersome for long messages since they must be split into small packets for transmission, then reassembled at the destination.

Significant considerations in the communication element are error rates, direction, speed, and setup time of the link. Typically speeds range from 60-100 bps to 50K bps with error rates of one in 10^5 bits [2]. In any case the communication element is a critical factor in a network.

2.1.4 NETWORK NODES

The nodes of a network are the intersection points (or termination points) of one or more communication links. Within this context a node may contain a computer system, a user terminal, another network, or a data concentrator.

Obviously a major component of a node is the computer system, (often referred to as a host). Computers in a network may be homogeneous (identical) or heterogeneous (different). The problems of program mobility, file handling and data translation are greatly reduced in a homogeneous network. On the other hand a heterogeneous network can offer complementary systems to the user, for example, high-speed production and interactive editing [28].

The important point in any node is the communication link; regardless of the contents of the node there must be some support for the communication lines. Minicomputers with their speed, reliability, flexibility, and low cost are the usual choice for this support. Employing a minicomputer as a front-end to the network relieves the burden of the network interface and protocol that would otherwise fall to the host computer system. In addition it tends to make the communications network independent from the host, thereby enhancing reliability [18].

As has been mentioned a node may be some type of user terminal. This embraces a wide variety of devices from a simple teletype to a remote job entry station [7]. Support for this terminal can come either from a host computer system or from the interface minicomputer. In any case this node is a consumer of network services in contrast to a host, which provides services.

A node containing a data concentrator is usually a major switching centre designed to multiplex many lines onto one or two. A typical application is for a concentrator to interface remote dial-up teletypes to the computer network [26]. They may also be used to distribute messages and work among several hosts.

The concept of a node being another network can occur in two forms. This network may be a subnetwork associated with the host computer at that node or it may be a viable network entity in itself. Obviously, to interface the networks, there must be some consideration given to the unique attributes of each [37].

In summary a node may be a consumer or supplier (or both) of network computing power. A consumer may be any user, host computer or network and a supplier may be a computer or a network.

2.1.5 LOGICAL STRUCTURE

The previous sections have discussed the topology, control, communications and nodes of a network. These items define the physical structure of the network, its composition, and its implementation. They do not, however, reflect the logical organization of the network. The above attributes describe how the network operates but not, for example, what the messages contain or their utility. The logical organization of a network defines the functional capabilities of the network, that is, what the user sees when he views the network.

Networks may be classified into four general categories:

(1) SPECIALIZED -networks which provide a specific service.

This type of network provides the user with a limited capability directed toward some specific type or class of problem. A typical example would be an airline reservation system or the file transport subnet of the OCTOPUS network [6]. In either case the user has a severely limited set of services he may request from the network.

(2) CENTRALIZED -networks which allow user access to a particular machine.

A network of this type is logically (and physically) structured as a tree with users as leaves and the host computers as roots. The sole purpose of this type of network is to assign users to machines. Host access can be explicit, the user requesting a specific host, or implicit, the network assigning the user to a host. In any case there is only minimal host-host interaction. TYMNET [3], CYBERNET [14] and GE Information Service [3] are typical networks in this category.

(3) DISTRIBUTED -the network is a single source of computing power.

In this environment host computers collectively cooperate through the network to provide service to the

user. The user in turn views the network as a single large computer facility. TUCC [24] and DCS [22] are examples of this organization.

(4) GENERAL -general purpose network.

This type of network does not have a designated logical organization; rather, it is very flexible in nature and can include any of the above structures. It is primarily a research tool to investigate network design and behavior. This is the case in the ARPA [5] and TSS [25] networks.

The logical structure of a computer network reflects the functional purpose of the network. Using topology, communication, control, and nodes as building blocks the designer structures the network to provide some functional capability to the user.

2.2 COMPUTER NETWORKS

2.2.1 CURRENT NETWORKS

The previous section has outlined some of the basic characteristics of a computer network. These attributes are reflected by some of the networks in use today.

(1) ARPA

The Advanced Research Projects Agency (ARPA) [1,4,5,]

network is an experimental computer network joining university and research centres. Its primary purpose is to investigate computer networks as well as provide service to the various research centres. It is composed of approximately 45 host centres which are located throughout the United States and elsewhere. These host centres are connected in a distributed fashion with at least two alternate paths between each host.

The communication section of the network forms an autonomous network on its own [15,18]. This network is independent and transparent to the host computer centre. It is composed of 9 and 50 kilobaud leased phone lines and a minicomputer, called an interface message processor (IMP), which performs message switching. Thus a host passes a message for a remote host to the IMP which then sends it to the remote host's IMP, thence to the remote host. The interface message processors maintain distributed control of the communications network [20,21]. Control is adaptive since messages are sent to intermediate IMP's along the shortest path (in time) to the destination IMP. This path changes dynamically with loading and errors in the communication lines. As well, maintenance and performance logging by the IMP's can be done from a remote location. The maximum message size is 8000 bits, which is broken into 1000-bit packets by the IMP for transmission throughout the

network. Since each packet may take a different route and arrive in random order the destination IMP reassembles the packets and gives the message to the host system. The maximum transit time for messages end to end through the network is 0.5 sec.

The nodes are composed of heterogeneous machines or user terminals. If a node consists solely of terminals a special terminal IMP or TIMP [19] provides the necessary terminal support. However, if the node contains a host computer, the local machine provides the necessary assistance. The host computers represent nearly every manufacturer and type, the most notable member being the ILLIAC IV [1]. A network control program at each host provides control and monitoring of individual programs associated with the network.

The ARPA network is by far the largest and best-known research-oriented network.

(2) DCS

The Distributed Computing System (DCS) [1,13,] is an experimental computer network being developed and built at the University of California at Irvine. Its purpose is to service small- and medium-scale computers and yet maintain: low cost for both start up and expansion, reliability, and

easy addition of new services.

To achieve these goals, a ring topology was chosen [23]. The ring consists of a circular communication path. Nodes are attached to the network through a ring interface. The processors are homogeneous minicomputers. Communication is accomplished by the ring interface acquiring an available message slot and placing the message on the ring. All messages are sent to processes rather than processors. Each ring interface reads any message sent to its processes.

Control is distributed throughout the network with a network support program in each processor [22]. Each node of the system specializes in some particular application. For example, one node may support disk files and dataset manipulation. When a job arrives at the network it is assigned to the node which can provide the most efficient service. A notary records the activity of services within the network. The unique communications structure facilitates the distributed logical organizations of this network.

(3) MERIT

The Michigan Educational Research Information Triad (MERIT) [1,10,11,12] network is a network effort by Michigan State University, Wayne State University and University of Michigan to share the computing services of the three

schools.

This is a fully connected but distributed network. Communication is through dial-up lines and minicomputer front-ends. In contrast to the previous networks this is a circuit-switched network. Minicomputers acting as network interfaces perform the necessary control functions. The host computers are two 360/67's and one CDC 6500.

> "The objective of the Merit network is to allow a pair of processes to be connected. The purpose of the connection is to allow records to be transfered between the connected processes. The connection is a duplex, logical path between a pair of processors. Only one connection is allowed between any pair of processes in the initial implementation ... a Master/Slave relationship is imposed on the connected processes so that the record traffic is manageable." [12,P65]

(4) TUCC

The Triangle Universities Computation Centre (TUCC) [1,24] network is a joint effort by Duke, North Carolina State and North Carolina Universities. The network topology is a star organization. The remote nodes communicate with the central node, which is also the major computer facility for the network. The remote nodes are 360 model 30's and 40's and the central node is a 360 model 75 with LCS. The remote nodes do some batch processing but are mainly engaged in transmitting jobs to the central processor. Since the network is composed of homogeneous machines, problems of

communications and control are greatly reduced.

(5) OCTOPUS

The OCTOPUS [1,6] network is an inhouse computer facility at the Lawrence Berkeley Laboratory. It is composed of three subnets each performing some network function. The first is a fully connected but distributed teletype subnet designed to connect terminals with time-shared hosts. The second is a centralized file transport subnet which provides the necessary file structure for the entire network. The third is a centralized remote job entry network. The nodes of the network are heterogeneous (CDC 6600's, 7600's, Sigma-7's and PDP-6's) computers with time-sharing operating systems. There are no direct host-to-host links. Rather, a user at a terminal, through the teletype network, is assigned to a host computer. A host computer uses the file transport subnet to access the user's files.

(6) CYBERNET

The CYBERNET [1,14] network is a commercial network which links Control Data Corporation's data centres. The purpose of the network is to provide reliability, load leveling, and better utilization of computing centre manpower as well as supplying customers with a wide selection of facilities, and shared programs and data bases.

CYBERNET is a distributed network in both topology and control. Communication is message-switched and circuit-switched. The network contains a wide variety of communication devices, including high-speed dedicated lines to slow-speed dial-up connections. A user terminal is connected to the communication system which, joins either a concentrator or a centroid. The concentrator is primarily a message-switching centre, which directs the user to the centroid which can provide the desired service. The centroids are the major workers of the network. They consist of CDC 6600's and 7600's. A centroid functions similarly to a concentrator except that it contains a major service computer.

This network is a typical example of a centralized logical organization. The purpose of the network is to allow user access to a computing centre, and for this reason, communication is user-to-host. By serving the user through the network it is relatively convenient to shift the user from centroid to centroid. Since the processors are homogeneous this is transparent to the user. Thus the network provides the reliability and flexibility required for load sharing.

(7) TSS

The TSS [8,1,25] network is a collection of homogeneous machines which are linked as a network. This network is sponsored by IBM and several of its customers, each with IBM 360/67's operating under TSS/360. Since each host of the network is identical, program and data sharing are relatively easy. In addition it is easy to modify and change the network since one change will alter all nodes.

The network is distributed in both topology and control. A network command interpreter interfaces the user (and thus the local host) with the remote host. Communication is through circuit-switched direct host-to-host data links. Control functions are resident in the TSS operating system. Although the network is homogeneous some nodes contain subnets.

(8) TYMNET

The TYMNET [3,26] network is a commercial network designed to provide interactive support to a remote user. The network topology is distributed but control is centralized. The network controller resides as a normal time sharing user on an XDS-940. There are three dormant controllers, one of which is awakened if the active controller fails. Although most of the hosts are XDS-940's

there are also some PDP-10's and Varian 620's. A user dials into a communication processor, a minicomputer, which provides support for the terminal. The communication processor has a message-switched, store-and-forward organization. The function of the controller is to establish a path or virtual channel through the network to the appropriate CPU. This path is fixed and provides low overhead, efficient communication. The host machines of the network are heterogeneous.

(9) GE Information Services

The General Electric Information Services network [3,27] is designed to allow remote user access to a computing centre. The user terminals are linked to a remote concentrator which performs the necessary buffering and support for the user's terminal. The remote concentrators are linked to a central concentrator which then interfaces to the computing facility or to a switching concentrator. The switching concentrator is linked to another central concentrator. The communication path between concentrators is dual 4800 or 9600 bps full-duplex lines. The communication philosophy is store-and-forward, message-switched. The central concentrators are responsible for directing the user's message to the computer system containing his ID and files. Control of the network

essentially resides in the central concentrator. These minicomputers monitor the traffic on a particular line and during network down time can reconfigure to improve utilization.

(10) IBM NETWORK/440

The IBM Network/440 [9,28,29] is a heterogeneous network of computers designed to study the problems associated with networking. It is principally located at IBM's J. Watson Research Center. The topology and control of the network is centralized. All network jobs are sent to the central controller. This controller interprets the user's network command language and sends the job to the appropriate machine. Communication is through 2,400 and 40,000 bps half-duplex lines. The communications subsystem at the central node acts as a store-and-forward digital switch for messages in the network.

(11) OTHERS

The above sections have summarized the significant characteristics and components of some of the major networks. There are also large numbers of lesser-known networks. Some of these are specialized networks designed for one specific function while others are research-oriented projects.

The National Association of Securities Dealers Automated Quotations (NASDAQ) System [3] is designed to give up-to-the-minute bid and asked prices of securities. Similarily airline reservation systems are designed to provide a conversational real time information system. These are only two examples of many such specialized networks.

Some research-oriented networks are the Star Ring system at the University of Toronto [32] and the MISS project [31] at the University of Chicago. The former is a communication ring which allows the interconnection of computers, peripherals, and terminal. The MISS project deals with providing a hierarchy of computer support for minicomputers in a laboratory environment.

There are of course many planned networks. Of these, CANUNET [30] is the major Canadian proposal under consideration. Its purpose is to promote Canadian industry in the development and manufacture of network components. The major users are to be Canadian Universities, hopefully reducing the cost of computer services. It is patterned very closely in hardware and software after the ARPA network in the United States.

2.2.2 NETWORK PROBLEMS

The previous sections have given a brief outline of the structure of a network and a review of some current networks. During that discussion some of the problems encountered in networking were noted. A more formal treatment of these problems will now be considered. They can be classified into three general areas: architecture, communication, and control.

(1) ARCHITECTURAL DESIGN

The architecture of a network is the specification of nodes and communication links between the nodes. The designer's goal is to construct the optimum network for a particular application [66]. The requirements of an interactive network are far different in terms of response time and message size from those of a remote job entry network. It is usual to specify some performance requirements in terms of message delay and a fixed maximum cost. The (optimum) network is then designed to meet these requirements.

The parameters which the designer must consider are, for example: cost, speed, error rate, and duplex/simplex connection in the communication link; setup time and node delay at each interface; message size, priority and

frequency of transmission, and routing in inter-node communications. It is generally agreed that the number of variables makes the problem unacceptable for integer programming analysis. Using heuristic algorithms some solutions for star and distributed topologies have been found [16,33]. Alternative approaches include statistical modeling and simulation to arrive at a suitable design [34,40]. However there is no general computationally efficient solution to the problem of network analysis or synthesis.

(2) COMMUNICATION

Communication is one of the most significant aspects of a computer network. Efficient operation of the communication process is critical to the performance of the network.

The most elementary consideration is a suitable communications protocol. This is the establishment of a prescribed method for acknowledgement of messages. In this regard the most fundamental question of error detection and handling arises. A message may be garbled or entirely missing. The problem then is to define an efficient strategy for acknowledgement and re-transmission of messages as well as how to handle the failing link.

In a store-and-forward, message-switched network it is

often necessary to break a long message into several small packets for transmission. This immediately raises the question of packet sequencing and message reconstruction. Since the interface message processor has limited memory capacity it is conceivable that several partially reconstructed messages are present but neither can be completed due to insufficient buffer space. Therefore a suitable algorithm must ensure that a message can be reconstructed correctly. In addition there must be some means of detecting duplicate packets or messages. This problem is two-fold; the first consideration is that packets may be duplicated by errors in the network and so exist in several different nodes only to arrive at random intervals. The second aspect is that, since the message identifier space is finite, there must be a means for distinguishing between different messages with the same identifier [18,19,21,37].

Efficient routing strategies and flow control are further questions pertaining to message switching. The requirements of these algorithms are to maintain a predetermined level of performance under normal loads while degrading gracefully with heavy loads. They are complicated by the "bursty" nature of tele-traffic and faults in the communication link. This problem has been formulated both as a heuristic algorithm [35] and adaptive flow control

[18,21].

Dealing with the actual communication links, work is being done on the allocation of channels to serve a variety of traffic having different message lengths and priorities [2]. In addition some studies [36] have been concerned with network reliability analysis. This is determining the reliability of the network from the communication failure rate. A question of considerable interest is the establishment of a suitable protocol for network to network links. More generally it is the problem of interfacing two heterogeneous networks [37].

(3) LOGICAL CONTROL

Logical control is the protocol which maintains the logical structure of the network. This is a problem concerning the interfacing of heterogeneous or homogeneous operating systems to promote symbiotic cooperation. As yet there is no suitable high level protocol to facilitate this relationship.

This problem may be split into three subproblems: (1) resource sharing, (2) data sharing, and (3) program sharing. Dealing with the first, there is no accepted viewpoint on either the mechanics or the structure of resource sharing. There is no agreement on what resources are shareable or how

they will appear and be controlled [15]. Nor is there any indication of an acceptable level of resource interaction.

Data sharing has two facets. The first is specification of data such that it is machine-independent. This is especially hard when accounting for machines of different word sizes and formats [39]. The second is file sharing. This notion allows host access to a remote file. It can be dynamic access in the case where the file remains resident at the remote station or it may simply be access to copy the file. In either case there are problems in translating between the data-management schemes of two systems [38].

The last section, program sharing, can also be split into two facets. The first is program sharing in the sense of a user using a program resident at some host. This implies implementing the necessary structure to allow the user to monitor and control the program. The second is program mobility. It is very rare that even a high-level program can be moved to another environment and still compile correctly. This is due to the idiosyncrasies of individual compilers. One approach is to edit the program when it is transfered [28] while another has been to provide a "universal" language [41,42].

CHAPTER 3

THE RESOURCE-SHARING PROBLEM

3.1 INTRODUCTION

In the context of computer networks there are a significant number of unanswered questions and unexplored possibilities. Previously some of the problems encountered in architecture, communication, and control were enumerated. Significant research efforts are under way in all these areas. The most notable results in both architecture and communication have accrued through the ARPA network project. However very little has been published concerning network control and resource management. Thus it seems reasonable to explore some of the questions encountered in network control, specifically resource sharing among nodes.

In its purest form, resource sharing is the integration of all computing resources into a single uniform entity. At this level, programs can be distributed across several machines, using those resources which suit their needs. More explicitly there are three areas of supervision that comprise an operating system: resource control, task management, and data management. To achieve true resource sharing these areas of supervision must be extended to include the network environment.

The obvious temptation in approaching network resource sharing is to design a "super network operating system". This is a monumental task! Such an integrated network system must provide the functions of a conventional operating system from an environment which is extremely complex. Consider the physical resources of the system; they are large in number and type, as well as being distributed throughout the network. These devices must be not only controlled but also allocated to programs in the network. Thus the system must coordinate the activity of distributed programs using distributed devices. The large numbers involved make this a difficult scheduling problem (not to mention deadlock). The situation is further complicated by the representation and translation of data types in a heterogenous environment. A problem of even greater difficulty is providing a system which is optimal for all user applications. Clearly if the user does not receive the "best" possible service from the network he will seek computing services elsewhere. These problems are undoubtedly solvable theoretically but the solution must be economic in terms of dollar cost and computational efficiency.

An alternative solution is to recognize the viability of an independent operating system and work with this concept. Therefore a solution lies in providing resource

sharing among network members but at a higher level than that which takes place in an operating system.

3.2 CURRENT RESOURCE-SHARING NETWORKS

Resource sharing in current computer networks is primarily directed toward allowing user access to a remote machine. Thus the process is usually user-initiated and controlled. There are three increasingly sophisticated forms of this activity.

The first is static resource sharing. This type of network primarily provides a computing service. The resource sharing consists of partitioning the computing service among the users. TUCC, CYBERNET and GE Information Services are examples.

The second is dynamic resource sharing. It is similar to the former category except that the partitioning or allocation of users is dynamic. OCTOPUS and Network/440 are typical examples.

The third type is the most sophisticated. In this category resource sharing is accomplished by allowing the user to access any node in the network. ARPA, TYMNET and TSS are examples. Thus the user can "sign on" to any machine in the network and utilize the resources of that system.

The DCS network accomplishes resource sharing in a slightly different manner. Its ring topology and small computer nodes promote specialization. This is similar to the specialized subnets in the OCTOPUS network.

The RSEXEC system [44] implemented on the ARPA network is the most sophisticated in terms of resource sharing. This system enlarges the range of resources available to a user to include those beyond his local system. By acting as an intermediary between the user and non-local systems the logical distinction between local and remote resources is removed. This system is restricted to those hosts which have the TENEX operating system.

In terms of resource sharing, the RSEXEC system comes closest to the ideal level of interaction. That is, a network which shares resources to solve the user's problem effectively and do so with a high degree of efficiency. However the RSEXEC concept must be extended to include heterogenous operating systems.

3.3 THE RESOURCE-SHARING PROCESS

Following is a proposed resource-sharing executive system for a computer network. It combines the idealized concept of intimate resource sharing with the practicalities of a network to achieve a greater degree of resource sharing

than is currently available.

3.3.1 A RESOURCE-SHARING EXECUTIVE SYSTEM

The executive system is based upon recognition that a network is composed of independent autonomous systems. Furthermore, the members expect to benefit from joining the network but recognize that they in turn must provide something to the network. As an entity the network is a source of resources which any node may call upon for help or service. In return each node is obligated to fulfill requests from the network. This is the basic resource-sharing process, the interchange and use of resources to mutual benefit.

Such a philosophy has many advantages. Each node is independent and the impact of the network is minimal. As a consequence there is no major change involved in the operating system. Furthermore the operating system is presumably optimal for the applications of its users and this will not be compromised. The operating system is the only entity which is aware of its capabilities and requirements. Therefore it should be the one which goes to the network for service. Resource sharing should be transparent to the user and under system control. This does not inhibit remote access or program sharing but it does provide a higher level of resource sharing. Other goals of

load leveling and enhanced reliability are also available within this context.

The function of the executive is to implement the network philosophy. The executive systen is necessarily distributed and resides as a module in the control program of each node. In addition to supporting remote access and program sharing the executive is mainly concerned with resource sharing.

In the executive system presented here there are many problems to be solved. For example, designing the interface between the operating system and the network, specification of protocols, a suitable data management scheme for the network, and resource sharing. It is this latter question which is considered by this work. Resource sharing is an important component of the executive and an understanding of this process is essential.

3.3.2 A RESOURCE-SHARING MODEL

It is apparent that more sophisticated resource sharing is the next feasible advancement in computer networks. However before any meaningful extensions can take place there must be a greater understanding of the fundamental process behind resource sharing. It is precisely that deficiency which this work fulfills. The intent is to

examine the concept of resource sharing; to formulate the theory which explains and predicts the performance of a network under resource sharing; and to relate these results to a network executive system. More correctly a resource-sharing network executive-system must account for the behavior of the network. Therefore it is necessary to have some foreknowledge of how the network operates in order to control it in a fashion which is reliable and optimal.

Unfortunately very little research has been published in this respect. The most notable work is the simulation of a network with respect to load leveling [45]. In this study three identical computer centers were simulated. To promote load leveling, job streams were equalized between the three systems. Under an asymmetrical load the objective was achieved.

Persuant to this question a model of a resource-sharing executive was constructed. This network model was then simulated and the performance of the model monitored. The behavior of the network was determined from these measurements.

Consider a network of n nodes. Each node is identical and consists of independent, autonomous computer facilities. They are independent in the sense that each node can operate equally well as a network host or as a private center. The

nodes are identical in that each has the same processing power or through-put capacity and a job can run equally well at any node in the network.

The executive system is one which views the network as a single source of computing power. A job submitted at a given node may run at any node in the network. Furthermore it is a distributed control system. Each node has embedded in its operating system a structure which allows it to interface to the network.

The function of the network is as follows. As jobs arrive at local nodes an attempt is made to run them there. If this is not possible the network is polled (in random order) to find a node which can accept the jobs. If there are no available machines the job remains in the wait queue until it can be run (at either a local or remote node).

The resource-sharing process has been modeled by a probabilistic structure. It is a study of the pure sharing of resources among several systems. The model chosen gives a good approximation to the real process (see Appendix A). The technique of modeling and simulation has been successfully used in many other applications. The parameters of the model are easily controlled and represent performance measurement characteristics.

3.4 SUMMARY

The resource-sharing executive was chosen to illustrate the function of the resource-sharing process in a computer network. This process was in turn modeled in order that it could be studied. In this manner several fundamental and important questions concerning resource sharing were investigated. Foremost was a study of the resource-sharing process to identify its behavior and significant parameters. Assertions which explain and predict this behavior were made. These results were related back to the resource-sharing executive system to permit efficient and reliable functioning of the network. From this investigation a greater understanding of resource sharing was achieved. The answers which it yields can be applied not only to future developments in network systems but also to systems currently in existence.

CHAPTER 4

RESOURCE SHARING

4.1 INTRODUCTION

In the previous chapter the model for a resource-sharing computer network was introduced. In this model, independent job streams arrive at each node in a network. The executive system shares the resources of the network among the jobs requiring service. Using this model several important points concerning resource sharing will be investigated.

4.1.1 THE RESOURCE-SHARING MODEL

As part of this investigation the network resource-sharing model was simulated. A more detailed discussion of the simulation is to be found in Appendix A. The items which the network shares are arbitrarily chosen to be measured in units called resources. Each node has a fixed total number of resources, all of which are considered to be identical. Removing the distinction between types of resources permits the behaviour of the resource-sharing process to be investigated.

The relevant parameters to the simulation are:

1. n   Number of nodes in the network.
2. a   Mean execution time of the jobs, defined by

an exponential distribution with mean a.

3. l Mean time between the arrival of jobs at a node, defined by a Poisson process with rate 1/l.

4. R_r The resource request size for each job defined as a uniform distribution on [1,100].

The length of simulation and sample interval may also be specified. The statistics collected from the simulation include:

1. U_T The mean number of resources in use at each node.

2. U^L The mean number of resources in use belonging to jobs which originated at the node at which it was run.

3. U_r The mean number of resources in use belonging to jobs which originated at a node other than that at which it was run.

4. Q The mean length of the queue of jobs waiting to be processed.

5. J The mean number of jobs processed by each node.

<u>Local utilization</u> (U^L) are those resources used by jobs which originate and execute at the same node. <u>Remote utilization</u> (U_r) are those resources which are used by jobs which originate and execute at different nodes. <u>Total</u>

utilization U_T is the sum of local and remote utilization and is the number of resources in use at any node.

$$U_T = U_L + U_r \tag{4.1}$$

4.1.2 LOAD ON THE NETWORK

Each node in the network has an independent stream of jobs arriving for service (according to a Poisson process). Each job has a certain resource requirement and execution time. Thus two nodes may have different job streams but similar loads. More explicitly, a large number of small fast jobs is equivalent to a small number of large slow jobs. The utilization of the machines is equivalent in each case. If R_T (a constant) is the total number of resources available at a node and R_r is the number of resources requested by each job then the expected number of jobs in execution at any time is

$$M = R_T / R_r.$$

Therefore the processing capacity of any node is M / a. To preserve the stability of the system the (expected) arrival rate must be less than the (maximum expected) processing capacity.

$$\frac{1}{l} < \frac{M}{a}$$

or

$$1 < (M \times l) / a. \qquad (4.2)$$

If $M=1$ this expression (4.3) reduces to the familiar stability condition for a single-server queueing system [46],

$$a / l < 1.$$

A measure of the load is

$$A = a / (M \times l) \qquad (4.3)$$

The load, A, is an indicator of the utilization of a node. Thus a load of 0 represents an infinite time between the arrival of jobs and hence no resource utilization. As the load approaches 1 the utilization of a node increases.

4.2 THE RESOURCE-SHARING PROCESS

4.2.1 PRINCIPLE OF NETWORK MULTIPLICITY

THEOREM 4.1

Given a network of resource-sharing computers, if some jobs must queue for service then:

a) the rate at which jobs are processed by the network is greater than the rate at which jobs are processed by a collection of independent machines under similar loads.

b) as the size of the network increases the processing rate also increases.

Proof:

Consider a collection of n independent nodes each with similar loads A, such that $A < 1$. There is a non-zero probability that any given node is idle,

$$\text{Prob(idle)} = P, \qquad 0 < P \leq 1.$$

Correspondingly, the probability of its being busy is

$$\text{Prob(busy)} = 1 - P.$$

An idle node is one which can accept another job and a busy node cannot.

In the collection of n nodes the probability of one or more nodes being idle is one minus the probability that they are all busy or

$$\text{Prob(at least 1 idle node out of n)} = 1 - (1-P)^n \tag{4.4}$$

Now combine these n independent nodes into a resource-sharing network. Consider the jobs which must queue for service. Since this is a resource-sharing network the probability that a queued job can run on another node is

$$\text{Prob(at least 1 idle node out of n} \mid \text{1 node is busy)} = 1 - (1-P)^{n-1}. \tag{4.5}$$

Thus, there is a non-zero probability that a queued job can begin execution sooner than it would otherwise be able to. Hence the processing rate is greater in a network than in an independent node situation (where the probability

given by 4.5 is zero).

Following a similar argument, add another identical node to the network. Again the probability that a queued job can run on another node is

$$\text{Prob(at least 1 idle node out of } n+1 \mid \text{1 node is busy)} = 1 - (1-P)^{n-1+1} \qquad (4.6)$$

Therefore in a network of size n+1 the chances of a queued job being executed sooner than in a network of size n is greater;

$$\text{Prob}(1 \text{ idle of } n+1 \mid 1 \text{ busy}) > \text{Prob}(1 \text{ idle of } n \mid 1 \text{ busy})$$

or

$$1 - (1-P)^{n} > 1 - (1-P)^{n-1}. \qquad (4.7)$$

Therefore the processing rate is greater in a larger network.

□

COROLLARY 4.1

If there are no queued jobs in a resource-sharing network then the processing rate is constant.

Proof:

A change in the rate of processing jobs is a direct result of there being some queued jobs available to be run on those node which periodically become idle.

□

COROLLARY 4.2

As the size of the network becomes large the processing rate becomes constant.

Proof:

Since the processing rate increases with increasing network size, eventually the network's capability will exceed the load placed on it. Recall that the load is constant and identical for each node. When this occurs there will no longer be any queueing jobs and from Corollary 4.1 the rate becomes constant.

□

There are two immediate consequences of the preceeding results. First, the total utilization is constant as the number of nodes in the network becomes large. This follows directly from Corollary 4.2. If the load and processing rate are both constant then necessarily the resources in use are also constant.

The second consequence is that local utilization approaches a constant as the number of nodes in the network becomes large. By definition total utilization is the sum of local utilization and remote utilization. It will be sufficient to show that for large n the remote utilization is constant.

a) Suppose U_r is increasing. From the Principle of Multiplicity this implies the probability of at least one node out of n being idle is increasing. However by Corollary 4.2 there are no queued jobs and the probability is one.

b) Suppose U_r is decreasing. This is impossible since the load is constant.

Therefore by contradiction the local utilization becomes constant for large n.

In summary the Principle of Multiplicity establishes the relationship between a network and its capabilities. It has been demonstrated that a resource-sharing network will have definite improvements over independent nodes and why this comes about. The relationship between total utilization, local utilization, load, and network size has been shown.

Proceeding with the investigation of the resource sharing process, several simulation runs were made. The parameters are summarized in Table 4.1.

	SIMULATION 1	SIMULATION 2	SIMULATION 3
1/1 =	45.0	28.1	45.0
R_r =	[10.0,30.0]	[2.5,22.5]	[2.5,22.5]
a =	50,100,120, 135,140,145, 150,160,175, 180,200,210	50,100,120, 135,140,145, 150,160,175, 180,200,210	80,160,192, 216,224,232, 240,256,280, 288,320,336

Table 4.1: Parameters for three simulation experiments.

4.2.2 FRAGMENTATION

As a member of a network, a host system is obligated to provide services to other nodes in the network. This necessarily entails the loss of certain resources to the network. Logically, the network becomes just another source of jobs to be processed by the node. However, a local user may be superseded by a network job. To this user the network is creating an artificial load depriving him of service. Considering the alternative view point, this infringement upon the local system has enhanced the capability of some other system. Membership in a network involves a balance between losses to the network of some resource and gain from the network of other services.

A measure of the involvement of a host computer center in a network is the fragmentation of the host's resources. Fragmentation is the portion of a local host computer system's (total) resources which are used by a remote system. In this discussion fragmentation refers to a state where the majority of the resources in use belong to remote jobs.

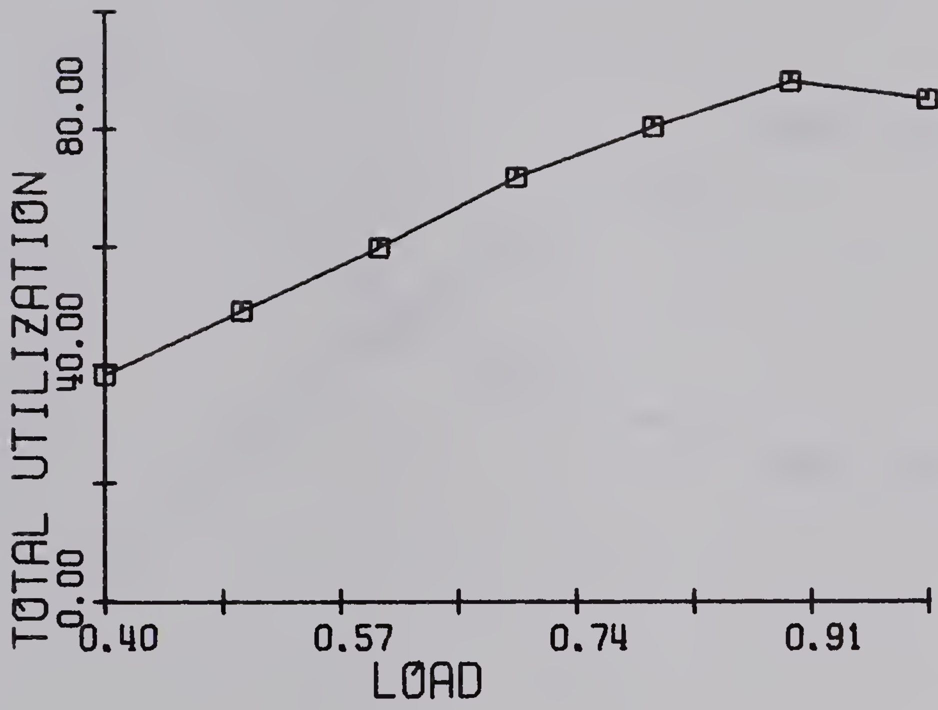

Figure 4.1: Total resource utilization in a five node network with varying load.

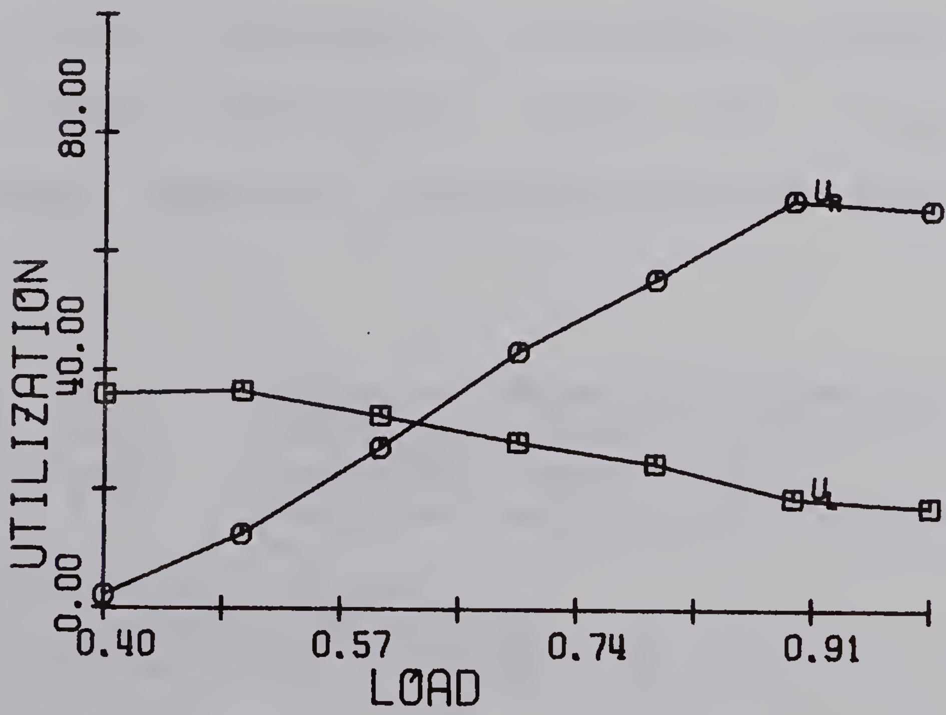

Figure 4.2: Resource utilization in a five node network with varying load.

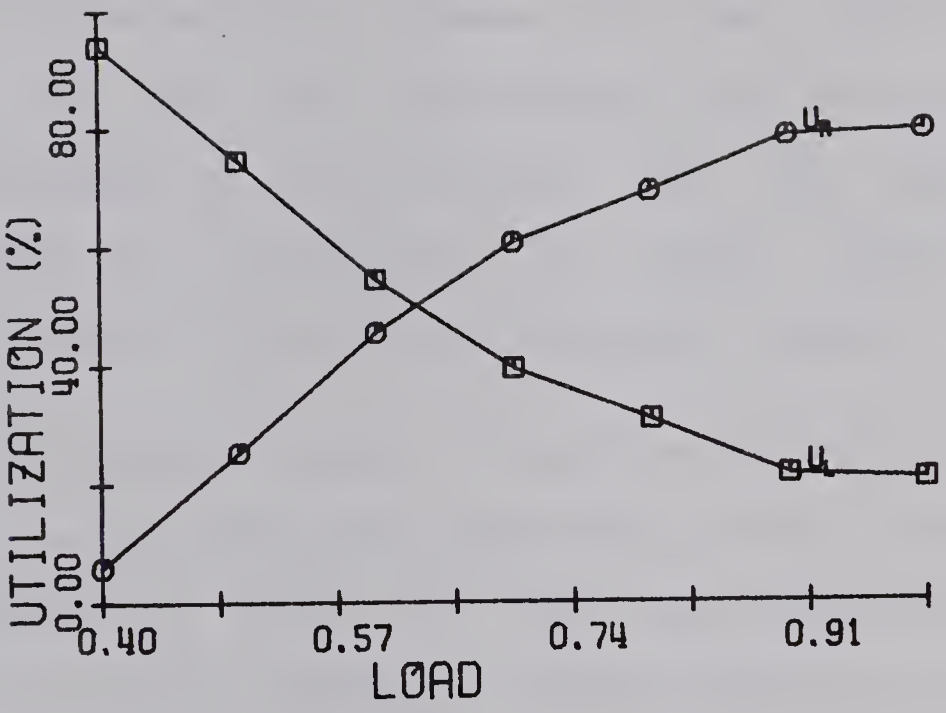

Figure 4.3: Percentage resource utilization in a five node network with varying load.

The initial experiment is designed to examine how the local and remote utilization change with respect to a varying load. Table 4.2 summarizes the parameters of this experiment.

n = 5	- Number of nodes in the network.
l = 0.5 sec.	- Interarrival time of jobs.
a = 1.0 sec.	- Execution time of jobs.
R_r= (10,70)	- Resources requests vary between 10 and 70.

Sample interval 1.0 sec.
Length of simulation 100 sec.
Loads simulated A = 0.4, 0.6, 0.7, 0.8, 0.9, 1.0.

Table 4.2: Parameters of a simulation to examine resource utilization under varying loads.

First consider total utilization as shown by Figure

4.1. As expected, as the load on the network nodes increases there is a corresponding increase in the utilization of resources. As the load approaches one the utilization reaches a maximum of approximately 90%. The relationship between load and utilization is linear. This will be explored in greater detail in a following section.

The two component parts of total utilization are shown in Figure 4.2. An unexpected phenomenon occurs. Consider the local utilization curve; as the load increases the local utilization actually decreases. Correspondingly the remote utilization increases.

This continues until A > 0.6; in this region the number of resources dedicated to remote jobs is greater than the number of resources dedicated to local jobs. That is, each node is processing more remote jobs than local jobs even though all nodes are similarily loaded. This fragmentation is even more apparent in Figure 4.3. In this figure the percentage of used resources belonging to remote and local jobs is shown. With a load of A = 0.6 the resources are divided equally between local and remote jobs; however, as the load increases the local utilization becomes less until at A = 1 only 25% of the used resources belong to local jobs. Since each node contributes 1/5 of the load as well as 1/5 of the computing power, it is expected that the local

utilization should be greater than is indicated by the simulation.

Resource sharing has degenerated, under a heavy load, to a situation where each node is using another node's resources to process its jobs. This domino effect is inefficient not only in terms of increased communications costs but also in processing time to redistribute jobs.

4.2.3 TOTAL RESOURCE UTILIZATION

The total utilization, U_T, for each of the three simulations is shown in Figure 4.4. Referring to these figures and the previous assertions, the total utilization is constant and independent of the number of nodes in the network. Any slight trends which appear may be attributed to the probabilistic nature of the simulation.

On the basis of this, the total utilization can be accurately modeled. Figure 4.5 shows the total utilization versus the load for each of the three simulations. There is a linear relationship between the total utilization and the load. Using the method of least squares [47,48] to estimate the parameters, the following relationship exists.

$$U_T(n,A) = 3.88 + 88.82 \times A \tag{4.8}$$

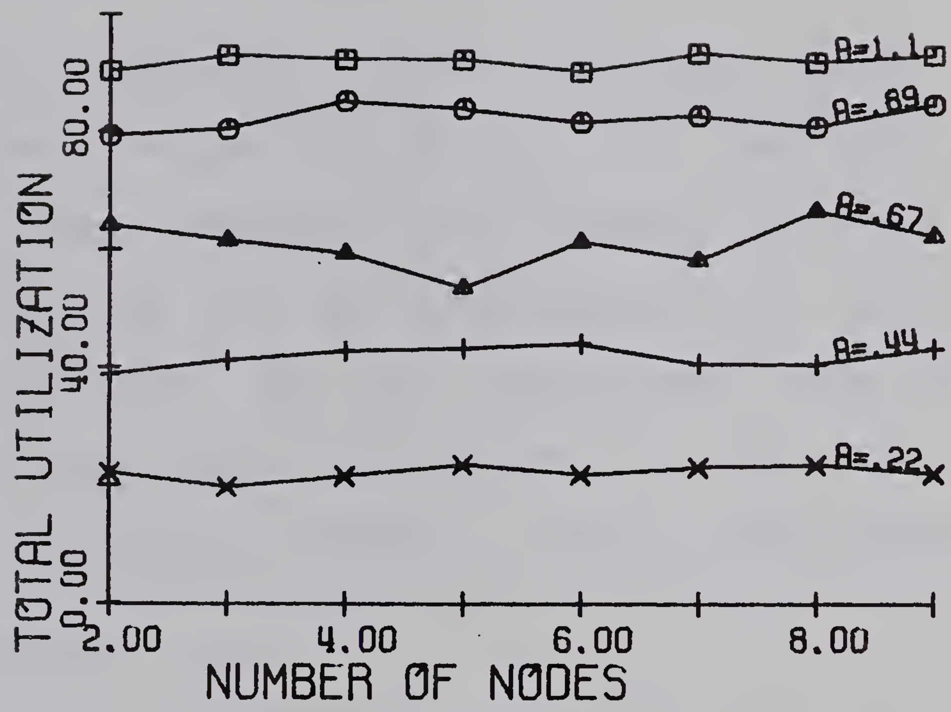

Figure 4.4: Total resource utilization in a multi-node network with varying load.

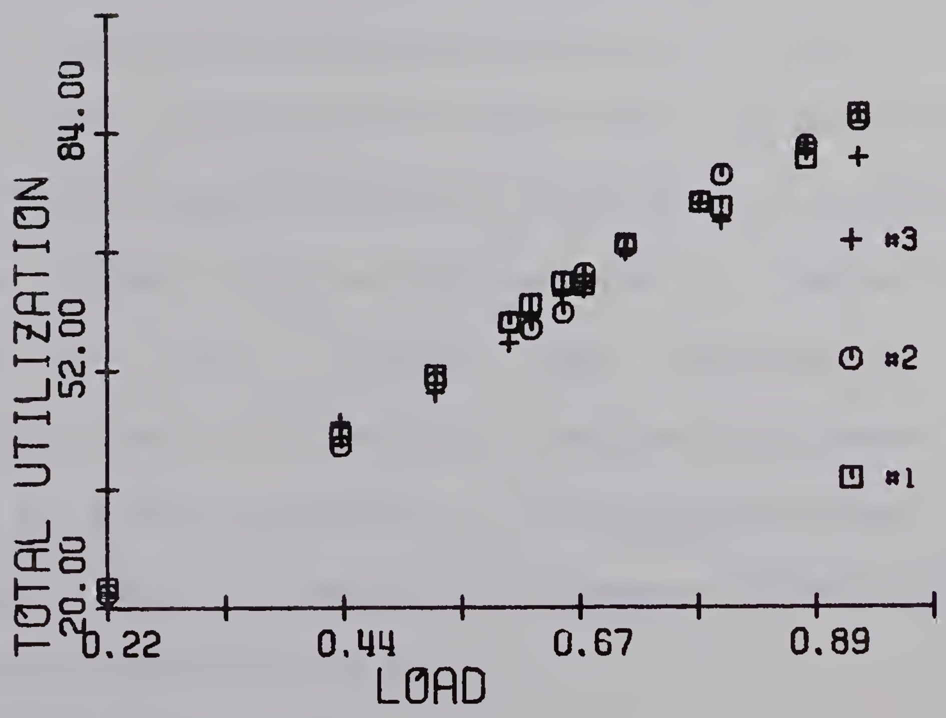

Figure 4.5: Total resource utilization versus load for a multi-node network.

4.2.4 LOCAL RESOURCE UTILIZATION

Again appealing to the previous work, the local utilization is a function of the load and the number of nodes in the network. The analysis of this aspect of resource sharing will be accomplished by observing that local utilization may be partitioned into four distinct types based upon load. These partitions are the states of the resource-sharing process in various environments.

1. LIGHT LOAD

In a lightly loaded network, that is A = (0,0.4], nodes are able to service requests immediately as jobs arrive. From Corollary 4.1 the local utilization is constant. Figure 4.6 shows the close proximity between the local utilization and the total utilization under these conditions; both are constant. This indicates that virtually no resource sharing is taking place nor is any necessary. However as the load increases (A = 0.4), observe the increasing disparity between U_T and U^L. Thus as the load increases it becomes necessary to share resources to maintain the same level of performance, that is immediate response. This class of local utilization is characterized by

1. Load A = (0,0.4]
2. A wait queue size of zero, indicating immediate response and adequate resources to handle the load.

3. Constant local utilization indicating each node is able to handle its own load.

Due to the fact that each node can handle most of its own load, indicated by the slight remote utilization, and the constant load on the network, the local utilization is

$$U^L(n,A) = C \qquad A = (0,0.4],\ n > 1. \qquad (4.9)$$

2. MODERATE LOAD

A moderately loaded network, A = (0.4,0.75], illustrates the increased performance which is available from increasing the size of the network. The transition between this class and the previous occurs when, for a small network size, a queue of jobs waiting to be processed exists. Table 4.3 gives an example of this transition.

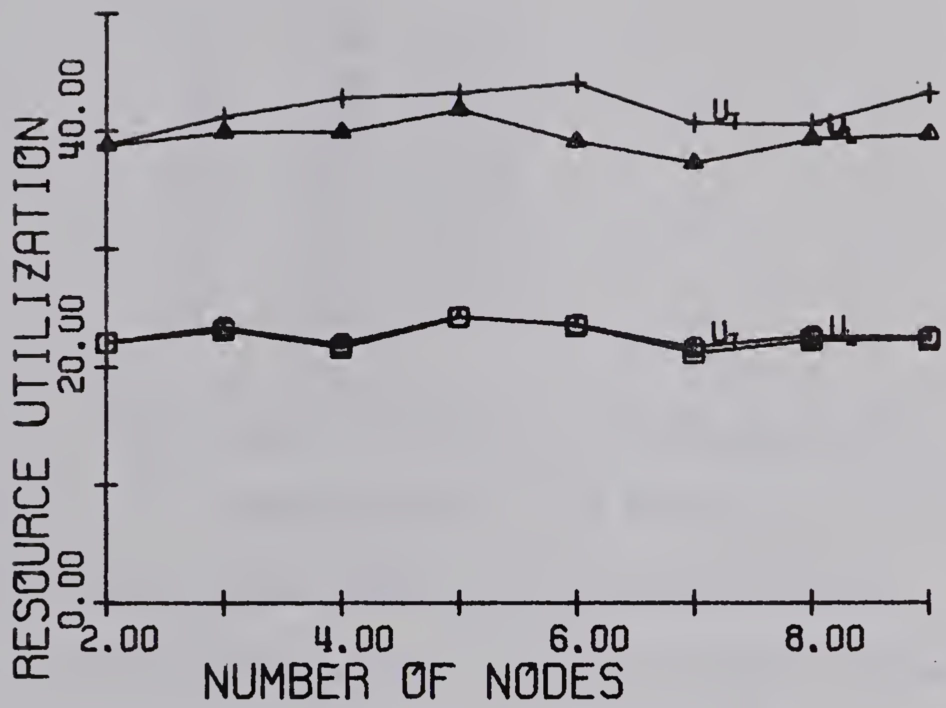

Figure 4.6: Local resource utilization in a multi-node network with a light load.

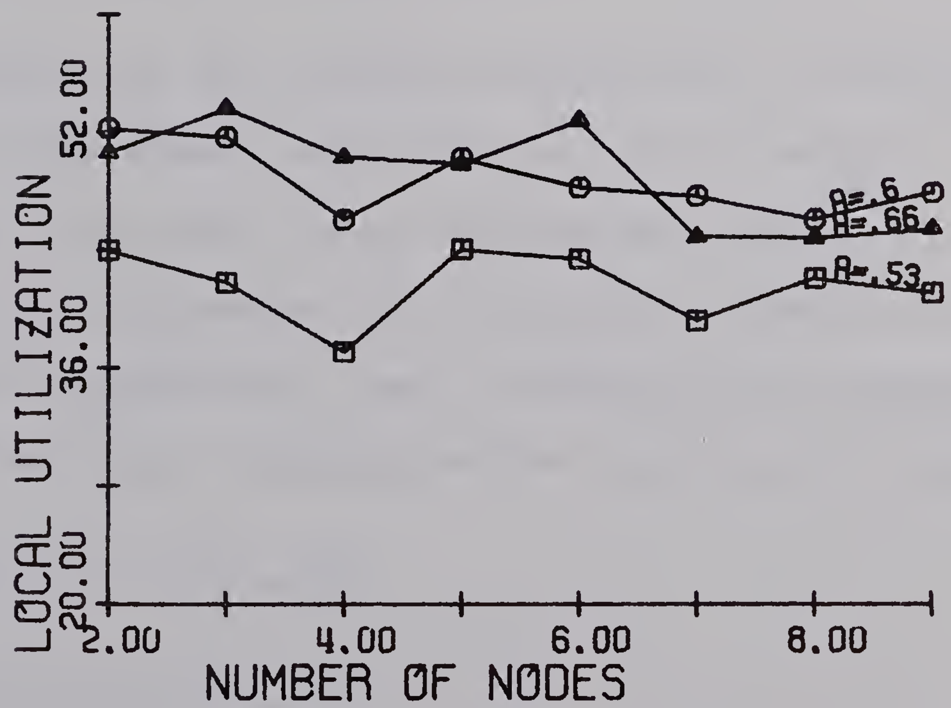

Figure 4.7: Local resource utilization in a multi-node network with a moderate load.

NODES	LOAD .63	.44
2	.16	0.0
3	.09	0.0
4	.03	0.0
5	0.0	0.0
6	0.0	0.0
7	0.0	0.0
8	0.0	0.0
9	0.0	0.0

Table 4.3: Wait queue size for a moderate load.

The Principle of Multiplicity and Corollary 4.2 show that for a larger network the queue size becomes small. Consider Figure 4.7. In all cases as the network size increases, the local utilization converges to a constant or steady state environment. Furthermore this convergence is from above and is defined in the following manner.

Referring to the previous assertions, by Multiplicity a network of sufficient size will be able to handle the load placed on it. When this occurs the network will enter a steady state environment and U^L will be constant. Let C be the local utilization for a steady state environment and $U^{L(2)}$ be the local utilization for a network of size 2. Then

$$U^{L(2)} \leq C + 1/2 \times \overline{\overline{R_r}}$$

$$U^{L(3)} \leq C + 1/3 \times \overline{\overline{R_r}}$$

$$\vdots$$

$$U^{L(i)} \leq C + [\,1/i \times \overline{\overline{R_r}}\,]k \qquad i=2,\ldots,n$$

where k=1 if the wait queue for the ith node is zero,

=0 otherwise.

Thus

$$U^{L}(n,A) = C + k[R_r / n]. \qquad (4.10)$$

The term 1/n arises in the following manner. Consider a job in the wait queue. When a job departs, the probability that a local job is run at the local node is 1/(number of nodes in network). Thus in a two-node network one-half of the waiting jobs will run locally.

Under a moderate load, a network containing few nodes will have a wait queue of jobs. As the size of the network increases the local utilization will monotonically decrease until a constant level of operation is reached, Equation 4.10.

At this juncture it should be pointed out that fragmentation as discussed in section 4.2.2 has not occurred. In the environments which have been examined so far the network has performed in a predictable and very desirable manner.

3. HEAVY LOAD

In a previous section it was shown that under a heavy load a resource-sharing network enters an unstable state called fragmentation. Generally a load in the region

A = (0.75,1] is sufficient to bring about this condition. Figure 4.8 shows the local utilization in this region. It is evident that as the network size increases, local utilization decreases until a minimum level is reached. Beyond this size, the level increases until a constant or steady state is reached. Table 4.4 shows the wait queue size and local utilization for various network sizes. The Principle of Multiplicity is confirmed by these results.

NODE	$Q(i)$	$U_L(i)$
2	0.26	51
3	1.03	43
4	0.16	42
5	0.14	39
6	0.02	28
7	0.0	35
8	0.045	39
9	0.0	39

$1/l$ = 45 sec.
R_r = [2.5,22.5]
a = 280 sec.

Table 4.4: Queue size and local utilization for a heavy load.

Observe that the size of the wait queue decreases as the network size increases. This indicates the increasing capability of the larger network. At a critical size of six nodes, the queue size becomes small and this corresponds to the minimum local resource utilization. Beyond six nodes, and as predicted, the local utilization rises.

4. OVERLOAD

The fourth and final condition is an overloaded state, $A = (1,\infty]$. Performance in this environment corresponds closely to the network in a heavily loaded condition in so far as fragmentation is concerned. If this load were applied to an independent machine there would be an unstable situation; however, by Multiplicity this is not the case.

In an overload situation, with complete fragmentation, the assignment of jobs to nodes is completely random. That is, a job will be assigned to any node with equal probability. If R_r is the mean size of the resource requests then the mean number of jobs in execution is

$$M = \frac{R_T}{R_r}.$$

Since local utilization is a product of the number of local jobs in execution and the size of their resource requirements, the local utilization is

$$U^L = M \times \frac{1}{n} \times R_r$$

or

$$U^L = \frac{R_r}{n}$$

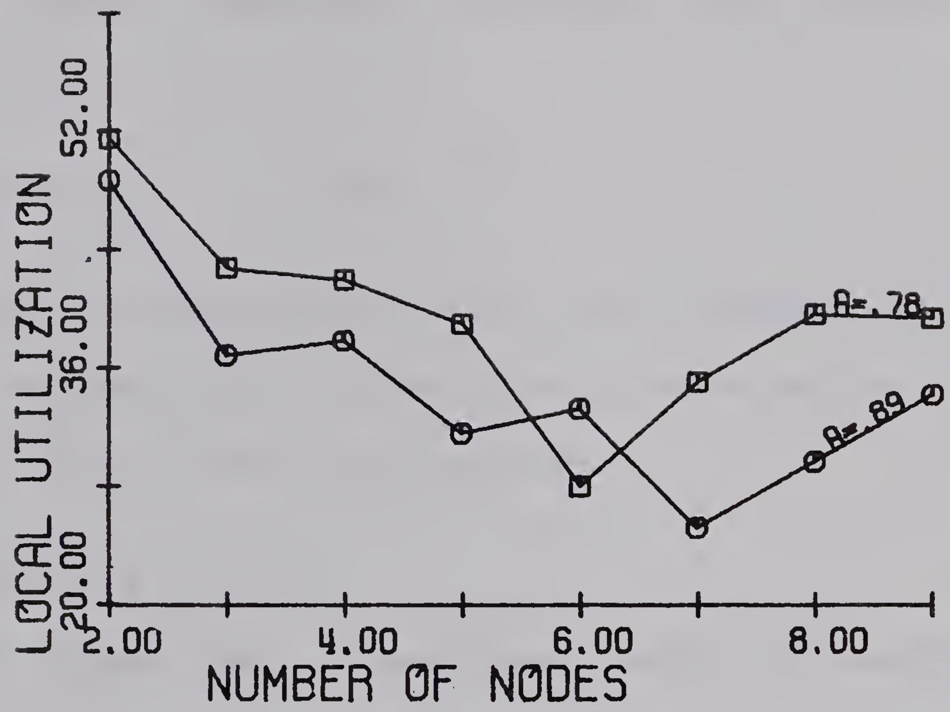

Figure 4.8: Local resource utilization in a multi-node network with a heavy load.

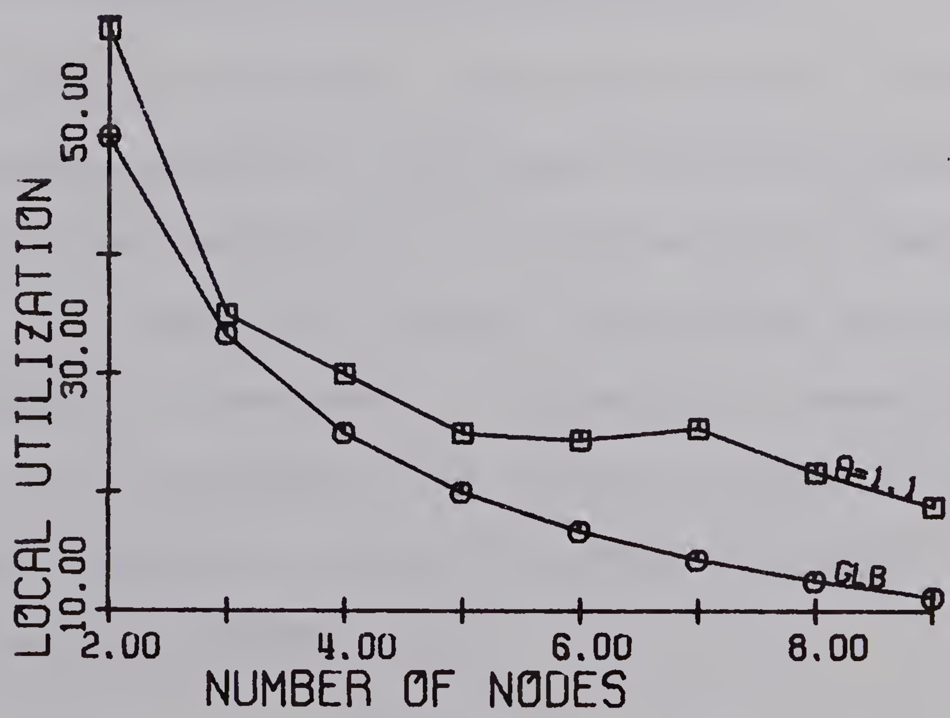

Figure 4.9: Local resource utilization in a multi-node network with an over-load.

Therefore under overload conditions the minimum local utilization is

$$U^L(n,A) = \frac{R_T}{n} \qquad A > 1. \qquad (4.11)$$

This of course assumes that $U_T=R_T$, all resource are in use. Under this assumption it is necessary to normalize the local utilization by the total utilization.

$$U^L = (R_T \times U^L) \;/\; U_T \qquad (4.12)$$

Figure 4.9 shows the normalized local utilization from a simulation with A = 1.11. The local utilization is decreasing and is bounded from below by 4.11.

4.3 RESOURCE ALLOCATION IN COMPUTER NETWORKS

The previous sections have developed a model for a resource-sharing network. This model has been explored both theoretically and empirically. On the basis of these results a problem in resource sharing occurs when the network is heavily loaded. Therefore, a potential resource-sharing executive for the network must include a suitable mechanism for resource allocation. Several algorithms will be explored in the following sections.

Using results from former sections, two types of resource-allocation strategies will be explored. The first

is a static control which governs resource sharing on an individual node basis. This control is distributed and can be applied independently by each node of the network.

The second is a dynamic control. The control parameter changes in time according to the performance of the network. It is based upon knowledge of the network's operation and applies to the network as a whole.

4.3.1 THREE RESOURCE-ALLOCATION STRATEGIES

It has been shown that in a generalized resource-sharing environment, a computer network is susceptible to the phenomenon of fragmentation. This arises from the resource-sharing process, and to preserve the efficiency of the network this situation must be controlled. Three resource-allocation schemes will be explored here.

The allocation strategies, uncontrolled, limited request, and limited acceptance, are algorithms suitable to a distributed-network executive system. The first is the benchmark or worst case against which the other strategies are compared. The other two correspond to the only places where some control can be applied -- the source of the resource request and the provider of the resources. In more detail the strategies are:

(1) Uncontrolled

The network is allowed to operate free of any controls.

(2) Limited Request

This algorithm attempts to control fragmentation by controlling the circumstances under which the local host will seek help from the network. It recognizes that network membership entails the dedication of some resources to network jobs. Control is achieved by setting a threshold which indicates the local host's involvement with network jobs. If the available resources plus those resources, above the threshold, belonging to a network job still cannot satisfy the local job, then a request is made to the network. The first available network host provides service. In essence, a request is made to the network only when the required service cannot be supplied.

(3) Limited Acceptance

The limited-acceptance algorithm controls resource allocation by refusing network jobs if the number of resources controlled by remote jobs will exceed some threshold.

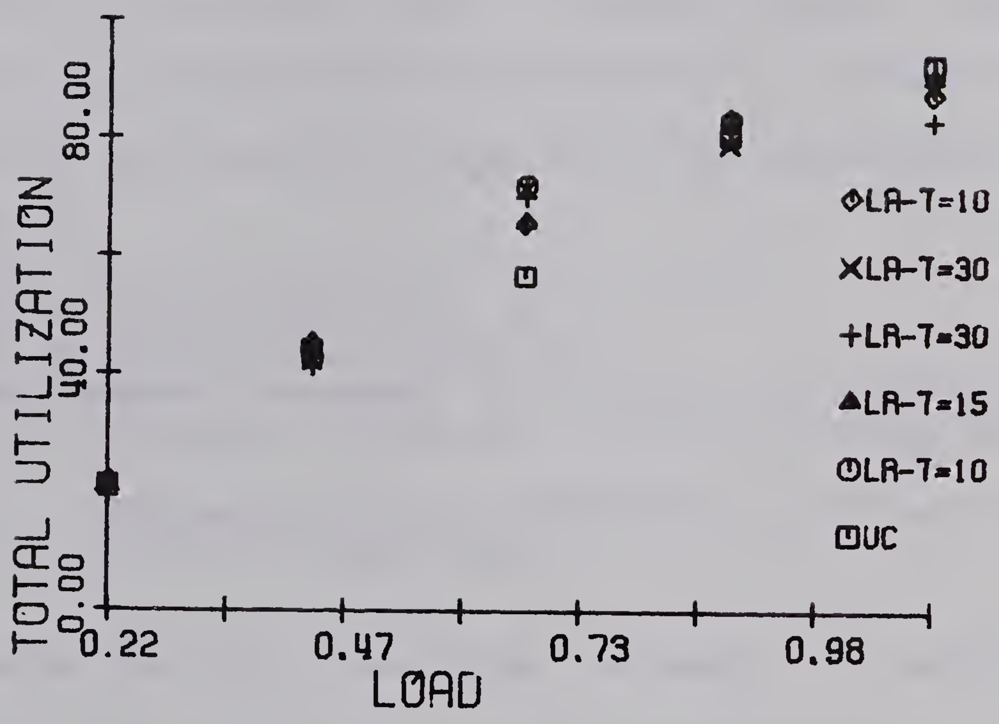

Figure 4.10: Total resource utilization for three resource control algorithms.

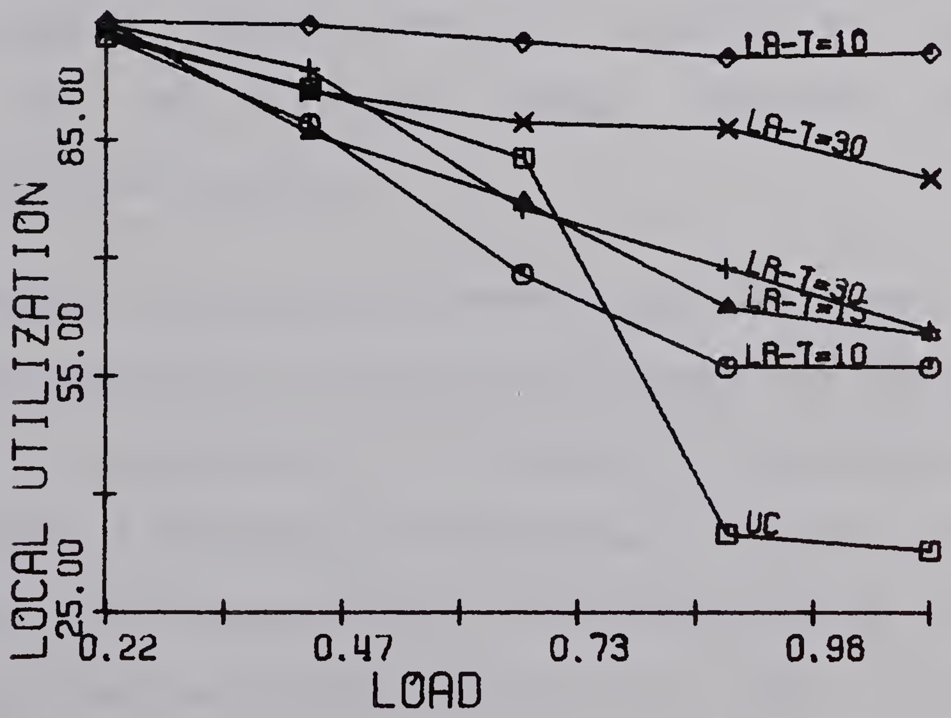

Figure 4.11: Local resource utilization (normalized) for three resource control algorithms.

To investigate these algorithms a network of computers was simulated using the above 3 control schemes. Table 4.5 summarizes the parameters to the simulation. The parameter T (resources) is the control variable of the algorithms.

Nodes = 5
Load A = 0.22, 0.44, 0.67, 0.89, 1.11,
Sample interval = 60 sec.
Length of simulation = 3600 sec.
Limited Request threshold T = 10, 15, 30 resources.
Limited Acceptance threshold T = 15, 30 resources.

Table 4.5: Parameters for a comparison of three resource control algorithms.

The three control algorithms attempt to modify the behavior of the network in such a manner that under heavy loads fragmentation is controlled. With reference to Figure 4.10 total utilization of the network resource is preserved by the two control schemes. This is significant since any reduction in this figure would indicate decreased performance by the network.

The normalized local utilization for all three schemes is given by Figure 4.11. Considering first the uncontrolled algorithm, at a load of A = 0.75 there is a discontinuity or sharp decrease in the local utilization. At this point the network can be assumed to be in an undesirable condition; for example, at A = 0.89 there is only 35% local utilization. By comparison the limited-request algorithm shows linear response to increasing load. However local

utilization is not extremely sensitive to the control parameter. It is an improvement; with a load of A = 0.89 the local utilization is greater than 55%. The limited-acceptance algorithm provides the best control. It is extremely sensitive to the control parameter as well as giving a graceful response to increasing load. At a load of A = 0.89 the local utilization is greater than 85%.

The major failing of the limited-acceptance scheme lies in the fact that it attempts to use the network only as a last resort. The control is structured such that the available resources of a node are inflated. These inflated resources are, of course, unusable and merely serve to force use of the network only when it is necessary. This clearly fails under a heavy load since a node is forced to turn to the network by the overwhelming size of the load.

In conclusion, an uncontrolled network begins fragmentation under moderate loading conditions and can be rejected outright as a feasible control. Limited acceptance provides for a remarkably linear increase in remote utilization. However the threshold parameter has only a moderate effect on the scheme, and fragmentation is still high. The best results accrued from the limited-acceptance algorithm

4.3.2 LIMITED-ACCEPTANCE ALLOCATION

Pursuant to the conclusions of the previous section the limited-acceptance algorithm is explored in greater depth. The simulation of a computer network with parameters shown in Table 4.5 is repeated. However the control threshold has values T=15,20,25,30,35, and 40.

Recall that this algorithm limits the amount of remote resource utilization to the threshold values. The results are presented in Figure 4.13. For each threshold value the algorithm provides robust control of remote utilization. Secondly the remote utilization responds in a linear fashion to increasing load. Using the method of least squares [47,48] to estimate parameters, the relationship is

$$U_r = 0.61 \times T - 6.03 \tag{4.13}$$

where U_r is the mean (normalized) remote utilization and T is the threshold value. Using this function the network may be "fine tuned" in response to some load.

Figure 4.12 presents the total resource utilization for the various threshold values. Again the total utilization is similar for all threshold values, indicating a stable control scheme.

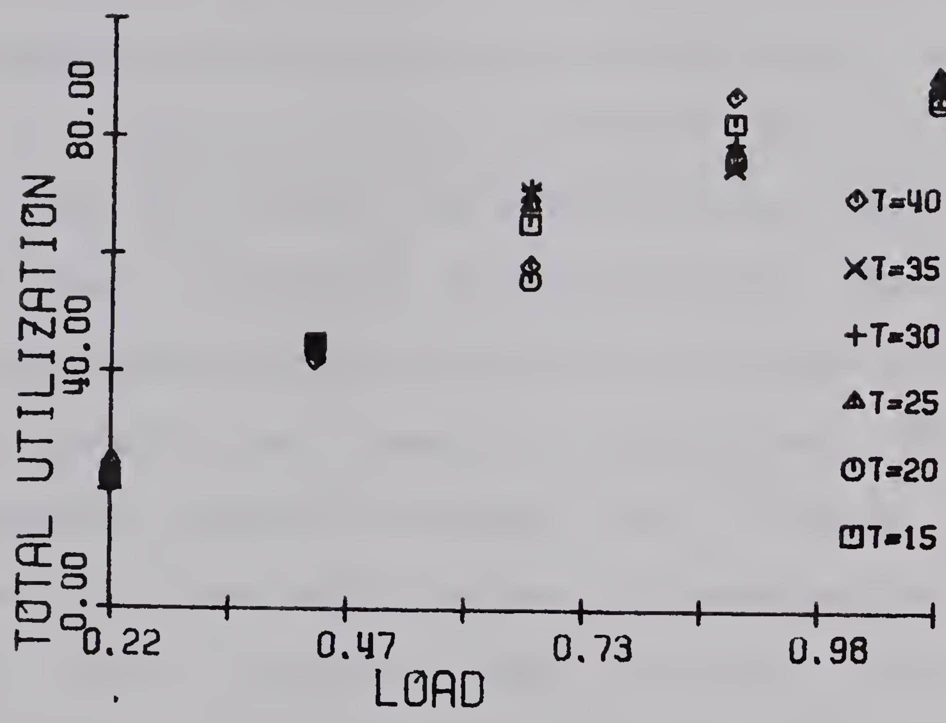

Figure 4.12: Total resource utilization for the limited acceptance resource control algorithm.

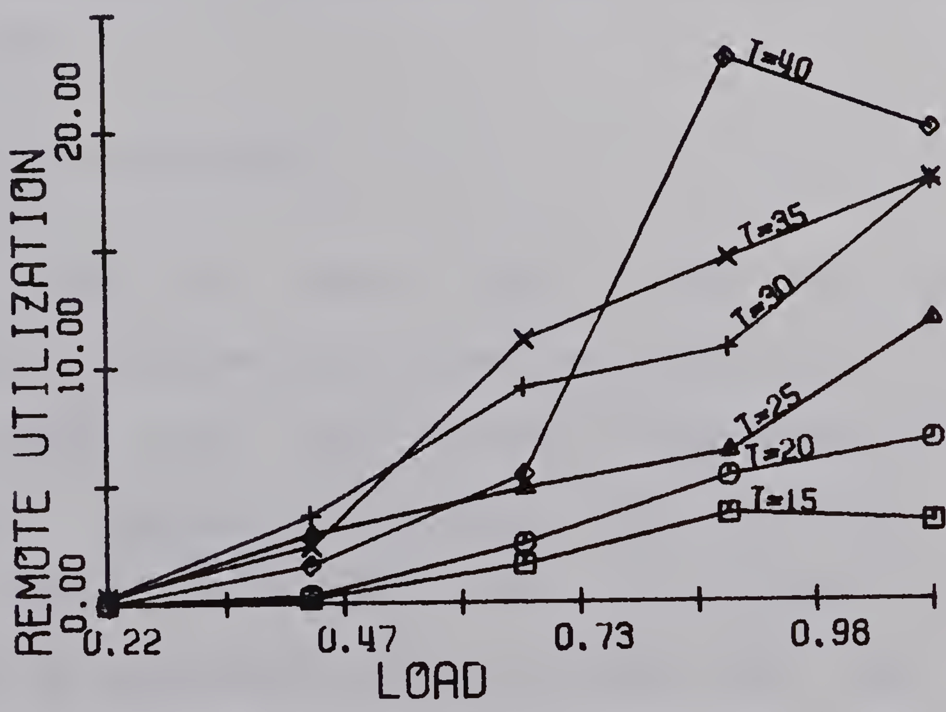

Figure 4.13: Local resource utilization for the limited acceptance resource control algorithm.

To summarize, the limited-acceptance resource-allocation strategy is desirable for several reasons. First, it is in keeping with the concept of independent nodes. The decision to accept or reject a network job is made by the node, which may (or may not) provide service. Thus the node is at all times in control of its operation. Secondly, the threshold parameter is very strong in the sense that it can effectively limit the resource utilization. Third, the linear relation between loading and remote resource utilization is beneficial since it provides for graceful degradation of a heavily loaded network. Finally, the function indicated by 4.13 clearly defines the relationship between the threshold and the mean remote utilization thereby providing a mechanism for accurately controlling fragmentation.

4.3.3 DYNAMIC ALLOCATION

It is known that under heavy loads the network is susceptible to fragmentation (see section 4.2.2). Control of this is based upon the following observation. Figure 4.8 shows local resource utilization for various sizes of networks under heavy loads. It is apparent that local utilization is inversely related to both the load and the number of nodes. For small networks the local utilization is higher than for large networks. This suggests a means of

controlling fragmentation based on the network size.

There are two forms in which this control may be applied. The first is restricting the size of the network to maintain a predetermined level of local utilization. Relating the number of nodes to the load will provide a good static control but necessarily a small network.

The alternative form is dynamically altering the network size. When local utilization falls below some threshold value the network size is decreased. As the local utilization rises the network size is increased. Control of the size can take place as a physical partitioning of a large network into semi-independent subsets or by restricting the number of nodes polled in an attempt to acquire resources.

Table 4.6 gives the parameters of a simulation of this latter scheme.

Load = 0.89
Nodes = 2,3,...,9
Threshold T = 50, 60, 70 resources.
Sample interval = 60 sec.
Length of simulation = 3600 sec.

Table 4.6: Parameters of a simulation for dynamic resource control.

The resulting local utilization for both controlled and uncontrolled is shown in Figure 4.14. This control scheme is very good for large networks. As can be seen it maintains a constant level of utilization for all network sizes.

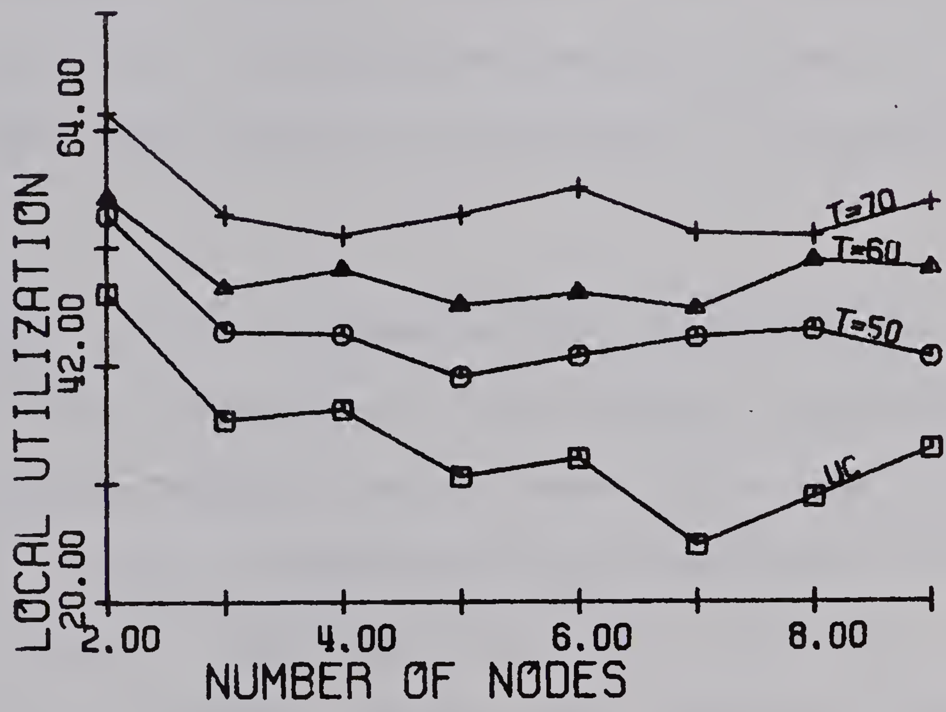

Figure 4.14: Local resource utilization for the dynamic resource control algorithm.

CHAPTER 5

CONCLUSIONS

5.1 SUMMARY OF RESULTS

The previous chapter presented the results of an investigation into the resource-sharing process in computer networks. From this, several significant conclusions have been drawn.

Foremost is the phenomenon of fragmentation. This has been shown to be a breakdown in the orderly performance of a resource-sharing computer network when it is subjected to a heavy load. Such an occurrence is consistent with similar phenomena observed in other computer systems. Consider thrashing in a paging system or response time in an overloaded operating system. Therefore a similar event has been identified for resource sharing in computer networks.

The second major result of this thesis is theorems which predict the performance of the computer network under varying conditions. These theorems show that a large network is more efficient and stable than a smaller one. Providing the load on the network is within a reasonable range the resource-sharing process is stable and uniform.

An empirical investigation confirmed the previous

theorems as well as revealing several important points. The relationship between resource sharing, network size, and load was determined. The performance of the network under varying loads revealed more information on the phenomenon of fragmentation.

Subsequent to these points control strategies which recognize the potential danger of fragmentation were investigated. Control is best implemented in the logical structure of a node which accepts resource-sharing requests from the network. An alternative scheme of control through the size of the network was also investigated. It can be applied in a dynamic fashion or as a static control built into the network during design.

5.2 APPLICATIONS OF RESULTS

The conclusions reached in this study have two immediate applications. The first is significant in the design phase of a network. It has been shown that a potentially unstable situation can exist. The assertions concerning network performance give the designer insight into the type of environment in which his system will operate. The second application is in control strategies for handling fragmentation. These mechanisms can be added to current networks .

5.3 SUGGESTIONS FOR FURTHER WORK

This study is one of the first investigations of resource sharing at this level in computer networks. As such it has laid the groundwork for more extensive future developments in computer networks.

The most obvious extension to this work is an extension of the resource-sharing model. There are several areas in which this could take place.

1. Employ an operating-system simulation which represents a well known system. Thus the input loads and measurement statistics would correspond to a "real world" situation.
2. A second extension would be diversification of the resource types. This is the differentiation between the type and availablilty of resources. Necessarily this suggests the need for more involved job streams which reflect the increased complexity of the resource classifications. Another fruitful addition would be to examine asymmetric loads and job types as to their effect on the network.
3. A third extension could be the investigation of different types of network organization.

This would include, for example, a study of different communication strategies and their effect on resource sharing, as well as the development of scheduling strategies and different control schemes. For example, consider a network of different node types. There is a need for scheduling and control investigations in this environment.

Another area of endeavor is a further consideration of the proposed resource-sharing executive system for a computer network. Within this area there are many problems to be considered.

1. Design of an executive system which could be embedded in an operating system with relatively little modification.
2. Specification of protocols for both communication and resource control. This includes some strategy for controlling and sharing various devices and programs.
3. Designing the operating system to network link.
4. Specification of a data-management structure to allow access and movement of files throughout the network.

Another problem area is to develop a queuing-theory model of the resource-sharing process. This entails investigation of the general G/G/N queuing system, where the number of servers, N, is a random variable.

In summary, these are general problems which have yet to be studied. The results of any of these areas would have immediate application to computer networks.

5.3 CONCLUSION

Computer networks serving a large population of users are entering the computing mileau. In the future such systems will play an even greater role in computer applications. The investigation of resource sharing is a significant and important advancement in network technology and the various aspects noted herein should be pursued.

BIBLIOGRAPHY

1. Farber, D.J., "Networks: An introduction.", Datamation, 18, 4(April 1972), 36-39.

2. Pyke, T.N. and Blanc, R.P., "Computer networking technology - a state of the art review.", Computer, (August 1973), 13-19.

3. Schwartz, M., et.al., "Terminal oriented computer communications network.", Proc. IEEE, 60, 11(November 1972), 1408-1423.

4. Frank, H., Kahn, R.E. and Kleinrock, L., "Computer communication network design - Experience with theory and practice.", Proc. of SJCC, 40, (1972), 255-270.

5. Roberts, L.G. and Wessler, B.D., "Computer network development to achieve resource sharing.", Proc. of SJCC, 36, (1970), 543-549.

6. Mendicino, S.F., "OCTOPUS: The Lawrence Radiation Laboratory network." in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 95-110.

7. Hobbs, L.C., "Terminals.", Proc. IEEE, 60, 11(November 1972), 1273.

8. Weis, A.H., "Distributed network activity at IBM.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 1-26.

9. McKay, D.B. and Karp, D.P., "IBM Network/440.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 27-44.

10. Herzog, B., "MERIT Computer Network.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 45-48.

11. Aupperle, E., "MERIT Computer Network: Hardware Considerations.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 49-64.

12. Cocanower, A., "MERIT Computer Network: Software Considerations.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 65-78.

13. Farber, D.J., "Data ring oriented computer networks.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 79-94.

14. Luther, W.J., "Conceptual bases of CYBERNET.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 111-146.

15. Kahn, R., "Terminal access to the ARPA computer network.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 147-166.

16. Frank, H., "Optimal design of computer networks.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 167-184.

17. Kleinrock, L., "Survey of analytical methods in queuing networks.", in Computer Networks, ed. R. Rustin, Prentice-Hall, Englewood Cliffs, N.J., 1972, 185-205.

18. Heart, F.E., et.al., "The interface message processor for the ARPA computer network.", Proc. of SJCC, 36, (1970), 551-568.

19. Ornstein, S.M., et.al., "The Terminal IMP for the ARPA computer network.", Proc. of SJCC, 38, (1971), 243-254.

20. Crocker, S.D., et.al., "Function-oriented protocols for the ARPA computer network.", Proc. of SJCC, 40, (1972), 271-279.

21. McQuillan, J.M., et.al., "Improvements in the design and performance of the ARPA network.", Proc. of FJCC, 41, (1972), 741-754.

22. Farber, D.J. and Larson, K.C., "The structure of the Distributed Computing System - software.", in Computer-Communications Networks and Teletraffic, ed. J. Fox, Polytechnic Press, Brooklyn, 1972, 539-546.

23. Farber, D.J. and Larson, K.C., "The system architecture of the Distributed Computer System - the communication system.", in Computer-Communications Networks and Teletraffic, ed. J. Fox, Polytechnic Press, Brooklyn, 1972, 21-28.

24. Freeman, D.N. and Ragland, J.R., "The response-efficiency tradeoff in a multiple-university system.", Datamation, 16, (March 1970), 112-116.

25. Rutledge, R.M., et.al., "An interactive network of time-sharing computers.", ACM Proc. of 24th National Conference, (1969), 431-441.

26. Tymes, L., "TYMNET - A terminal oriented communication network.", Proc. of SJCC, 38, (1971), 211-216.

27. Hench, R.R. and Foster, D.F., "Toward an inclusive information network.", Proc. of FJCC, 41, (1972), 1235-1242.

28. Hobgood, W.S., "Evaluation of an interactive - batch system network.", IBM Systems Journal, 12, 1(1973), 2-15.

29. Mckay, D.B. and Karp, D.P., "Protocol for a computer network.", IBM Systems Journal, 12, 1(1973), 95-105.

30. Twyver, D.A. and Dettwiler, W., "A network access facility for CANUNET.", University of British Columbia Computing Centre, (March 1973).

31. Ashenhurst, R.L., "The minicomputer interfacing support system project.", ICR Quarterly Report NO. 33, University of Chicago, Sec. A(May 1972).

32. Potvin, J.N., "The Star-Ring system of loosely coupled digital devices.", CSRG-7, (April 1971).

33. Frank, H., et.al., "Optimal design of centralized computer networks.", Networks, 1, 1(1971), 43-58.

34. Kleinrock, L., "Analytic and simulation methods in computer network design.", Proc. of SJCC, 36, (1970), 569-579.

35. Frank, H. and Chou, W., "Routing in computer networks.", Networks, 1, 2(1971), 99-112.

36. Van, R. and Frank, H., "Network reliability analysis: Part 1.", Networks, 1, 3(1971), 279-290.

37. Cref, V. and Kahn, R., "Towards protocols for internetwork communication.", private communication.

38. Fredericksen, D.H., "Describing data in computer networks.", IBM Sytems Journal, 12, 3(1973), 257.

39. Donaldson, M., et.al., "Preliminary draft: Proposed MICIS standard for data description.", Michigan Interuniversity Committee on Information Systems, (December 1970).

40. Frank, H., "Topological considerations in the design of the ARPA computer network.", Proc. of SJCC, (1970), 581-587.

41. Orgass R.J. and Waite, W.M., "A base for a mobile programming system.", CACM, 12, 9(September 1969), 507.

42. Waite, W.M., "Building a mobile programming system.", Computer Journal, 13, 1(February 1970), 28-31.

43. Streeter, D.N., "Centralization or dispersion of computing facilities.", IBM Systems Journal, 12, 3(1973), 283.

44. Thomas, R.H., "A resource sharing executive for the ARPANET.", Proc. of SJCC, 42, (1973), 155-163.

45. Bowdon, E.K., Mamrak, S.A. and Salz, F.R., "Simulation - a tool for performance evaluation in network computers.", Proc. of SJCC, 42, (1973), 121-131.

46. Kendal, D.G., "Stochastic processes occurring in the theory of queues and their analysis by the method of imbedded Markov Chains.", Annals of Mathematical Statistics, 24, (1965), 338-354.

47. Larson, H.J., Introduction to Probability Theory and Statistical Inference, John Wiley and Sons, N.Y., 1969.

48. McCalla, T.R., Introduction to Numerical Methods and Fortran Programming, John Wiley and Sons, N.Y., 1967.

49. Coffman, E.G. and Denning, P.J., Operating Systems Theory, Prentice-Hall Inc., Englewood Cliff, N.J., (1973).

50. Cox, D.R., and Lewis, P.A., The Statistical Analysis of Series of Events, Methuen and Co. Ltd., London, 1966.

51. Hemer, J.E., Heying, D.W., "Performance modeling and empirical measurements is a system designed for batch and time sharing users.", Proc. of FJCC, 35, (1969), 17-26.

52. Bryan, G.E., "JOSS: 20,000 hours at a console - a statistical summary.", Proc. of FJCC, 31, (1967), 769-777.

53. Ng, N. and Chum, W.Y., "On scheduling algorithms.", private communication.

54. Gaver, D.P., "Probability model for multiprogramming computer systems.", JACM, 14, 3(July 1967), 423-438.

55. Kimbleton, S.R., "Performance evaluation - a structured approach.", Proc. of SJCC, (1972) 411-416.

56. Rosin, R.F., "Determining a computer center environment.", CACM, 8, 7(July 1965), 463-468.

57. Hanssman, F., Kistler, W. and Schultz, H., "Modeling for computer center planning.", IBM Systems Journal, 10, 3(1971), 305-324.

58. Sreenivasan, K. and Kleinman, A.J., "On the construction of a represenative synthetic workload.", CACM, 17, 3(March 1974), 127-133.

59. Lucas, H.C., "Performance evaluation and monitoring.", Computing Surveys, 3, 3(September 1971), 79-91.

60. Wood, D.C. and Forman, E.H., "Throughput measurement using a synthetic job stream.", Proc. of FJCC, 39, (1971), 51-56.

61. Bucholz, W., "A synthetic job for measuring system performance.", IBM Systems Journal, 8, 4(1969), 309-318.

62. Campbell, D.J. and Heffner, W.J., "Measurement and analysis of large operating systems during system development.", Proc. of FJCC, 33, (1968), 903.

63. Seraphin, D.S., "A fast random number generator for IBM 360.", CACM, 12, 12(December 1967), 695.

64. _Mathematical Methods for Digital Computers_, ed. A. Ralston and H.S. Wilf, John Wiley and Sons, N.Y., 1967, 255.

65. Kuo, F.F., "The Aloha broadcasting packet communications system.", Proc. of CIPS-ACM Pacific Regional Symposium, (1974), 459.

66. Collier, R.J., "Network design factors.", Proc. of CIPS-ACM Pacific Regional Symposium, (1974), 449.

APPENDIX A

SIMULATION OF A COMPUTER NETWORK

INTRODUCTION

As an aid to studying computer networks a simulation program was written. The primary intent of the simulation is to examine the dynamic behavior of the network, specifically, job movement and resource utilization.

THE BASIC SIMULATION

The simulation consists of generating a stream of jobs at each node in the network. These jobs are then assigned to some node for execution. The state of each node is monitored at regular sample intervals. At the termination of the simulation the statistics which were collected are summarized.

The program, written in ALGOL-W (without the GOTO), is composed of a series of modular procedures which emulate the required operations.

The information collected on each job includes:

Arrival time -the time at which the job arrived.

Start time -the time at which the job began executing.

Stop time - the time at which the job stopped executing.

Resources - the number of resources required by the job.

Source node - the node at which the job originated.

Run node - the node at which the job executed.

At each sample interval the information collected on each node includes:

Wait queue - the number of jobs waiting to be executed.

Local resources - the number of resources in use by local jobs.

Remote resources - the number of resources in use by remote jobs.

In addition the total number of jobs processed by each node is kept.

Parameters to the simulation are:

1. Type of Simulation - which resource allocation scheme is to be used.
2. Intermediate Output - controls the

printing of statistics on the operation of individual nodes; otherwise, just the mean results are given.

3. Number of Nodes -the number of nodes in the network.
4. Threshold Value-(optional) the resource-utilization threshold required by an allocation scheme.
5. Four Parameters -parameters for the distributions: mean time between the arrival of jobs at a node; mean execution time of a job; and the interval (a,b) of the resource request size.
6. Sample Interval and Maxtime -the time between samples and the length of the simulation.

Note: items 3 to 6 are repeated as necessary until the number of nodes is less than one.

The results returned by the simulation are:

1. Each of the above parameters is displayed.
2. The mean wait queue size, mean number of locally used resources, mean number of remotely used resources, mean number of

jobs processed, and the total number of jobs processed are displayed.

The items in 2 above may be repeated for each node or may be the mean result from all the nodes.

Jobs are assumed to arrive at the nodes according to a Poisson process. Thus the interarrival time between jobs has an exponential distribution. The <u>residence time</u> of a job, that is (stop time)-(start time), is also exponentially distributed. The number of resources which a job requires has a uniform distribution on the interval (a,b) where $1 \le a \le b \le 100$. If R is a uniformly distributed random number on [0,1] then uniformly and exponentially distributed numbers may be generated by the following transformation [64]:

1. uniform distribution on (a,b)

$$S = a + (b - a) * R$$

2. exponential distribution -parameter L

$$S = - \ln(R) / L$$

A random number on [0,1] is generated by the multiplicative congruential method [63]. The choice of these distributions will be discussed in a subsequent section.

THE DETAIL SIMULATION

Essentially the simulation consists of 4 parts: (1) the arrival of jobs at a node, (2) assigning a job to a node for processing, (3) the departure of a job from a node, and (4)

sampling the performance of each node. The first and third sections are reflected by the data structure of the simulation. As jobs arrive at each node they are appended to the list of jobs waiting to be processed--the wait queue. Similarly when a job begins execution it is removed from the wait queue and placed, in order of departure (stop) time, in the list of currently executing jobs--the process queue. Both the wait queue and process queue are global to the network. All jobs from all nodes reside in either queue in time sequence. The second and fourth sections are the active elements of the simulation.

Since the network is sampled at regular intervals the simulation proceeds in these time steps. If h is the length of the sample interval then the job arrivals which occur in (t,t+h] are generated. The wait and process queues are examined and those job initiations and departures which can take place in this interval are handled (in order of their occurrence). When these events are exhausted the network is sampled and the algorithm repeated for (t+h,t+2h].

Processing proceeds as follows. Considering the wait queue, process queue, and the resources available at each node the next <u>feasible event</u> is determined. A feasible event is either the initiation of a job or the departure of a job from the network. A departure is always feasible. However

this is not true of initiations since a node may not have the resources to run the job. Therefore the list of waiting jobs is examined to find the first job which may be executed at some node in the network (not necessarily the node at which the job arrived). Thus several large jobs may be passed over in favor of a small job which can be run immediately. This strategy is chosen to maximize the utilization of the network. Clearly a more optimal strategy could be found using the resource size, estimated execution time, and currently executing jobs; however, such a discussion is beyond the scope of this thesis. The algorithm for assigning jobs to nodes is termed the <u>resource-allocation strategy</u>.

There are 3 possible outcomes from the above selection:

1. No Feasible Events -There are no departures nor any jobs in the wait queue which can be run.
2. Either a departure or initiation but not both is possible.
3. Both a departure and initiation are feasible and therefore the earliest of the 2 events becomes the feasible event.

There a two consequences of this selection process:

1. If there are no feasible events then the state of each node is sampled and the algorithm repeated for the next clock cycle.
2. Process the feasible event by removing the job from the process queue and releasing its resources or move the job from the wait queue to the process queue and acquire the necessary resources; continue with the next feasible event.

The selection of a feasible initiation event proceeds as follows. The list, in order of arrival, of all jobs in the network waiting to be processed is examined. An attempt is made to assign the first (oldest) job of the queue to run on the node at which it arrived. If this is not possible the network is polled to find a node at which the job may be run. In determining whether a job may be run at a remote node the specified resource-allocation strategy is used. If the job is still unable to run it is passed over and the next job examined. This procedure is repeated until either a feasible job is found or the list exhausted.

The first feasible departure event is always the head of the process queue since this list is sorted in order of

stop time.

Once a job has been selected for initiation the start time for the job is either the departure time of the last job (the job which released enough resources to run this job) or the arrival time of the job. The latest of these two times is the start time. It is assumed that task initiation is instantaneous. The stop time is computed by adding the generated residence time to the start time.

After the network's operation has been simulated for the specified time, the statistics which have been collected are summarized.

DISCUSSION OF THE ARRIVAL AND DEPARTURE PROCESSES

In the simulated model of a computer system the arrival and departure of jobs is assumed to take place according to some distribution. Specifically the length of time between the arrival of successive jobs and the length of time between the departure of successive jobs has some distribution. Thus in the simulation the inter-arrival and inter-departure times are generated according to some random process. The following discussion will attempt to justify these processes.

Poisson Process

In a computer system the arrival of jobs to be processed may be viewed as the occurrence of random events in time. The decision of a user to submit a job is completely independent of a similar decision by another user. Furthermore, the discrete nature of computers and channels makes it impossible for two or more jobs to arrive at exactly the same instant. Priority arbitration between channel interrupts also precludes the possibility of simultaneous arrivals. For these reasons jobs are assumed to arrive at a computer according to a Poisson process.

The properties of a Poisson process [49,P146], [50,P17], [47,P120] with parameter L are:

1. Given a short interval of size h
   a. the probability of 0 events occurring in h is 1 - L * h + o(h).
   b. the probability of 1 event occurring in h is L * h + o(h).
   c. the probability of 2 or more events occurring in h is o(h).
2. The occurrence of an event in one interval of length h has no effect on the occurrence or non-occurrence in another

(non-overlapping) interval of length h.

3. The probability of the occurrence of an event in (1) above does not change in time.

It is well known that computer usage varies in time with peak loads occurring during the afternoon. Clearly this violates property (3) of a Poisson process. However considering some reasonable interval of time, say a 1 or 2 hour period, it is plausible to assume that the arrival and departure rate is constant.

From references [50,P22] and [49,P148] if random variable X is the length of time from the occurrence of an event until the occurrence of the next event in a Poisson process with rate L then X has an exponential distribution with parameter L. The distribution function of X is

$$F(t) = 1 - \exp(-L \times t) \qquad (t \geq 0)$$

And the density function is

$$f(t) = L \times \exp(-L \times t) \qquad (t \geq 0).$$

The expected value is 1 / L and variance $1 / L^2$. Therefore in a Poisson process the mean time between events is 1 / L.

Arrival and Residence Distributions

The assumption that jobs arrive at a computer system in accordance with a Poisson process would seem, from the

preceeding discussion, ostensible. This is in fact born out by observation [51,52]. In these references the authors indicate that the inter-arrival time of jobs are exponentially distributed but give no statistical basis for this conclusion. Table 1 below summarizes the inter-arrival time for jobs during a 1 hour period at the University of Alberta Computing Center [53] in 1971.

Time (min.)	0.5	1.0	1.5	2.0	2.5	3.0	3.5	≥3.5
No. of Jobs	39	24	12	2	2	0	2	1

Table 1: Inter-Arrival Time of Jobs.

Under the hypothesis that the inter-arrival times are exponentially distributed the Chi-square statistic is 9.33 with 6 degrees of freedom. With a 1% chance of error the hypothesis is accepted.

Unfortunately it is not possible to make quite as strong a statement about the departure process. Recall that the residence random variable is the length of time that a job, once it begins executing, will take before departing from the system. This time includes the execution time as well as the time the task is blocked waiting for I/O or service. Once again appealing to the independence of jobs the completion of a job is completely independent of other jobs. That is, analogously to the arrival process, a

departure is the occurrence of a random event in time.

Several authors [54,55] have attempted to model computer systems under the assumption that a process alternates between ready, computing and blocked states. The length of these cycles is defined by an exponential distribution. This assumption is also borne out by empirical results [56]. Thus in the simulation the departure random variable was chosen to be exponential.

RESOURCE REQUEST

The resource request random variable is the number of resource units which a job requires. Resources include a wide variety of hardware and software items. For example, CPU time, channels, disk space, tape drives, card readers, line printers, core, as well as compilers, loaders, run time libraries, operating system services--the list is very long. At the most intricate level a simulation must account for the performance of a real computer system. Every resource must be modeled, its size, the length of use, the number of times it is used as well as exceptional conditions which arise from its operation. Clearly this is a difficult problem.

The problem of generating some reasonable resource utilization process is equivalent to the problem of

constructing benchmark or synthetic programs [58,59,60,61]. The question is not how to design the programs but rather whether they perform similarly to a real job. As pointed out in reference [59] there is not even agreement on what characterises job types.

It may be possible to simplify some of the above points by simulating only CPU time and I/O requests. However this necessitates some consideration of the service rate, number and type of channels. Since this simulation deals with a network the problem is duplicated for each node.

Essentially such a detailed simulation is not required to answer the point in question. As has been pointed out [62] a simulation should reflect the question it is to answer. The following generalization has been made to expedite the simulation. Each node or computer system will consist of 100 resource units. These units represent all the available resources of a computer--CPU time, core, peripherals and software modules. Each job places a request for some number of resource units. There is no distinction made between types or classes of resources or jobs. By accepting a job the system has decreased its ability to provide service by some finite amount.

Under this assumption it is recognized that every job in an operating system requires some portion of the system's

resources. Furthermore these resources are lost in the sense that while assigned to that job they can be used by no others. Alternatively it may be argued that an operating system is finite and under any operating conditions there are a finite number of jobs which the system can "digest" at one time. Consider OS/360 (MFT); it can support only a fixed number of partitions and at any time there is a maximum number of jobs in execution.

This approximation allows the simulation to proceed without having to become involved in the problems of current research in other areas. By normalizing the resources in this manner the question of job types and program performance which are necessary to represent the load on a system are ignored. Finally, it is no longer necessary to provide a detailed simulation of the internal interactions of an operating system.

CONCLUSION

The purpose of this simulation is to study the movement of jobs in a computer network. Necessarily this entails the modeling of the operating system and hardware at member nodes of the network. The broad question which this study examines permits a first order approximation to the "real" operation of the network. The choice of arrival and departure process which this simulation uses is well founded

in both empirical results and theoretical modeling. Normalization of the resources circumvents the problems encountered in performance measuring and load definition. Therefore the results of this simulation are at least a first order approximation to the performance of an actual system.